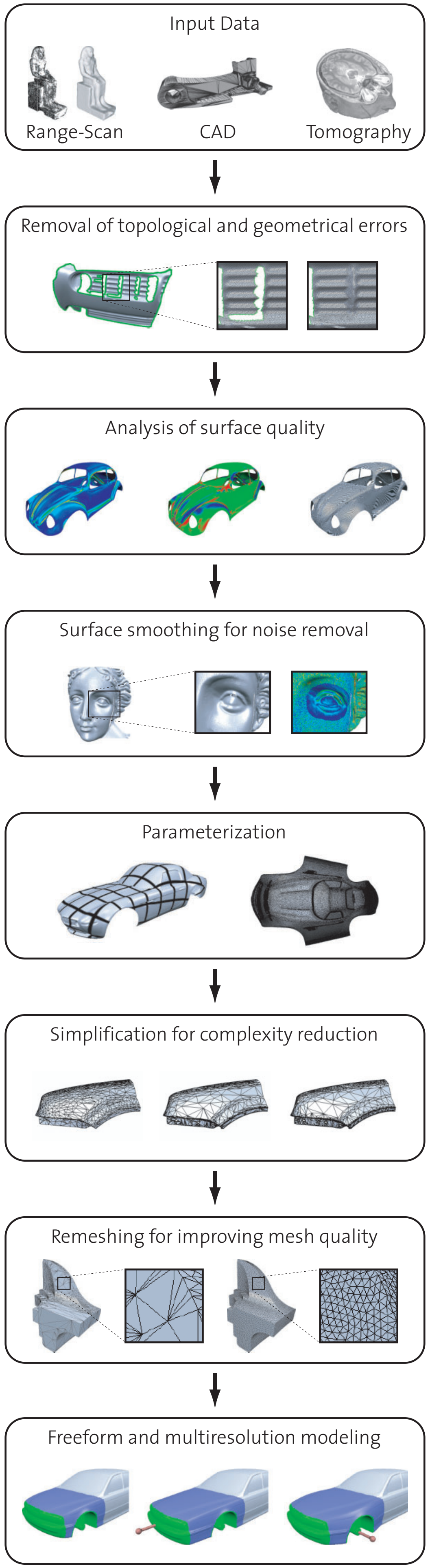


Efficient Solvers for (sparse symm. pos. def.) Linear Systems

Mario Botsch
ETH Zurich

SIGGRAPH2006

Problems in Geometry Processing

- Generic formulation as a PDE
  – Based on partial derivatives
- Discretization for triangle meshes
  – Finite elements / differences
  – Lead to linear systems (typically 10^4 to 10^6 DoFs)
- Partial derivatives are local operators
  – Sparse linear systems

Problems in Geometry Processing

- Most often the PDE can be considered as the Euler-Lagrange equation of an energy minimization problem

- or $\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}$ emerges as the normal equation for a least squares problem

➡ Systems are usually symmetric and pos. definite

Problems in Geometry Processing

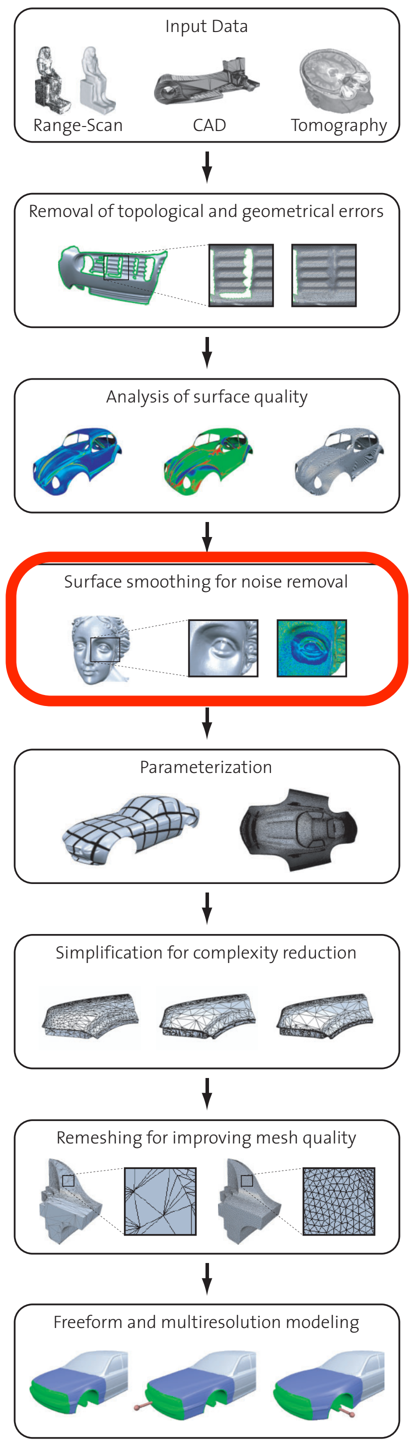
- Linear problems:
  - Solve $\mathbf{Ax} = \mathbf{b}$
- Non-linear problems:
  - Solve sequence of linear systems $\mathbf{A}_k \, \mathbf{x}_k = \mathbf{b}_k$
- Matrix **A** typically is
  - large
  - sparse
  - symmetric positive definite

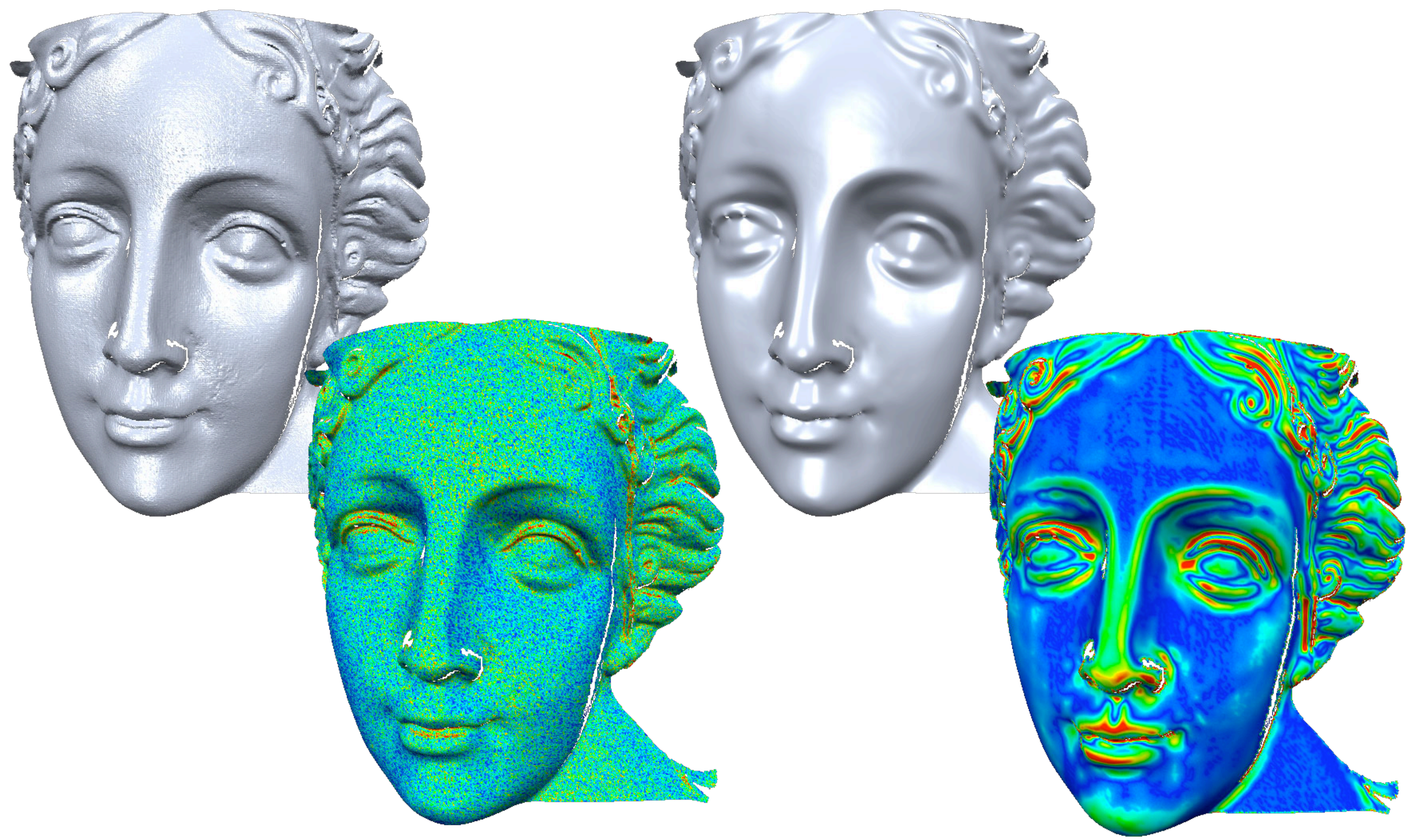
Non-spd systems:
See course notes

Overview

- **Application scenarios**
- Linear system solvers
- Benchmarks

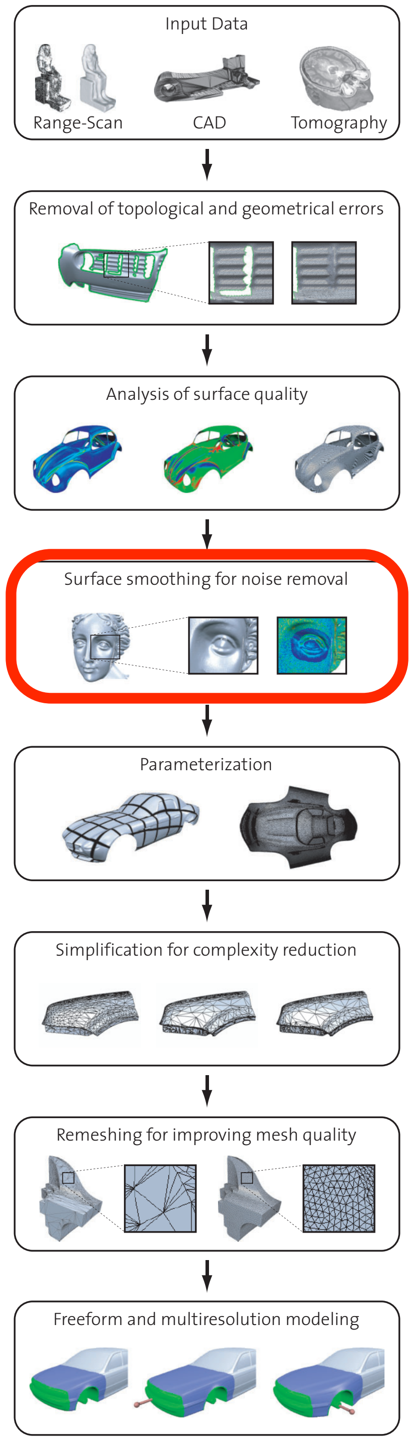
Implicit Fairing

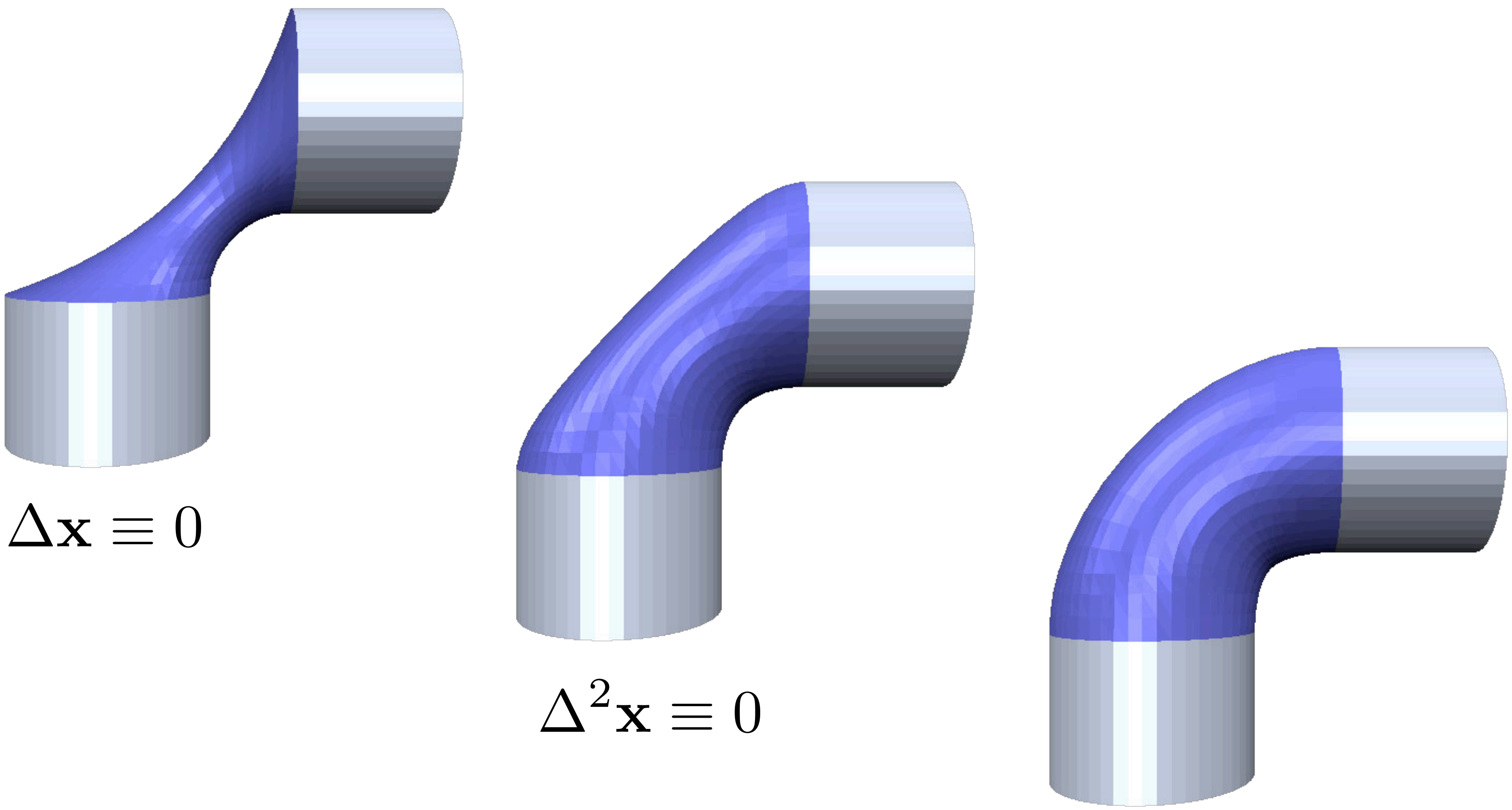




$$\left(\mathbf{I} \pm \Delta_S^k\right) \mathbf{x}_{k+1} = \mathbf{x}_k$$

Variational Energy Minimization



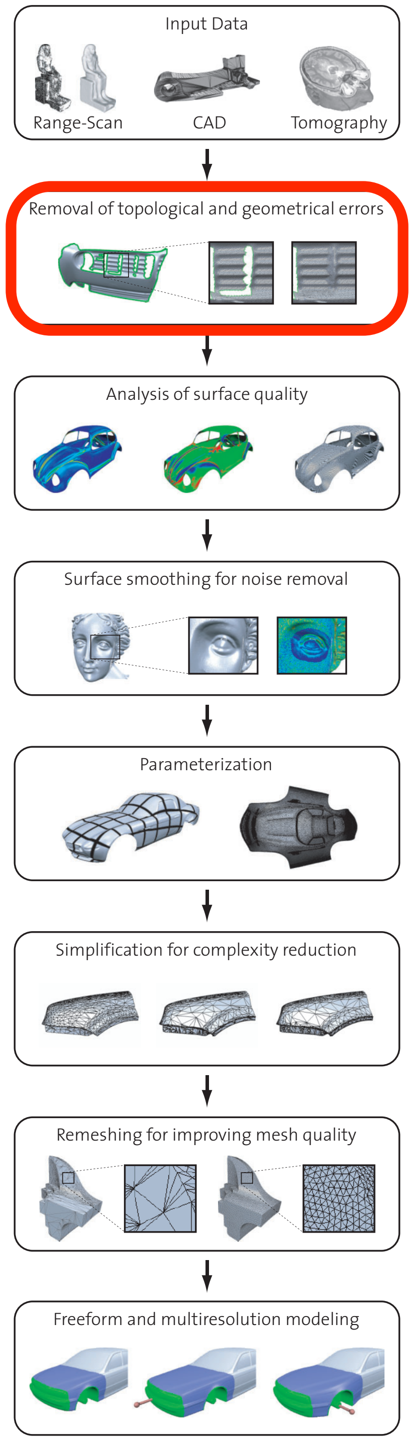


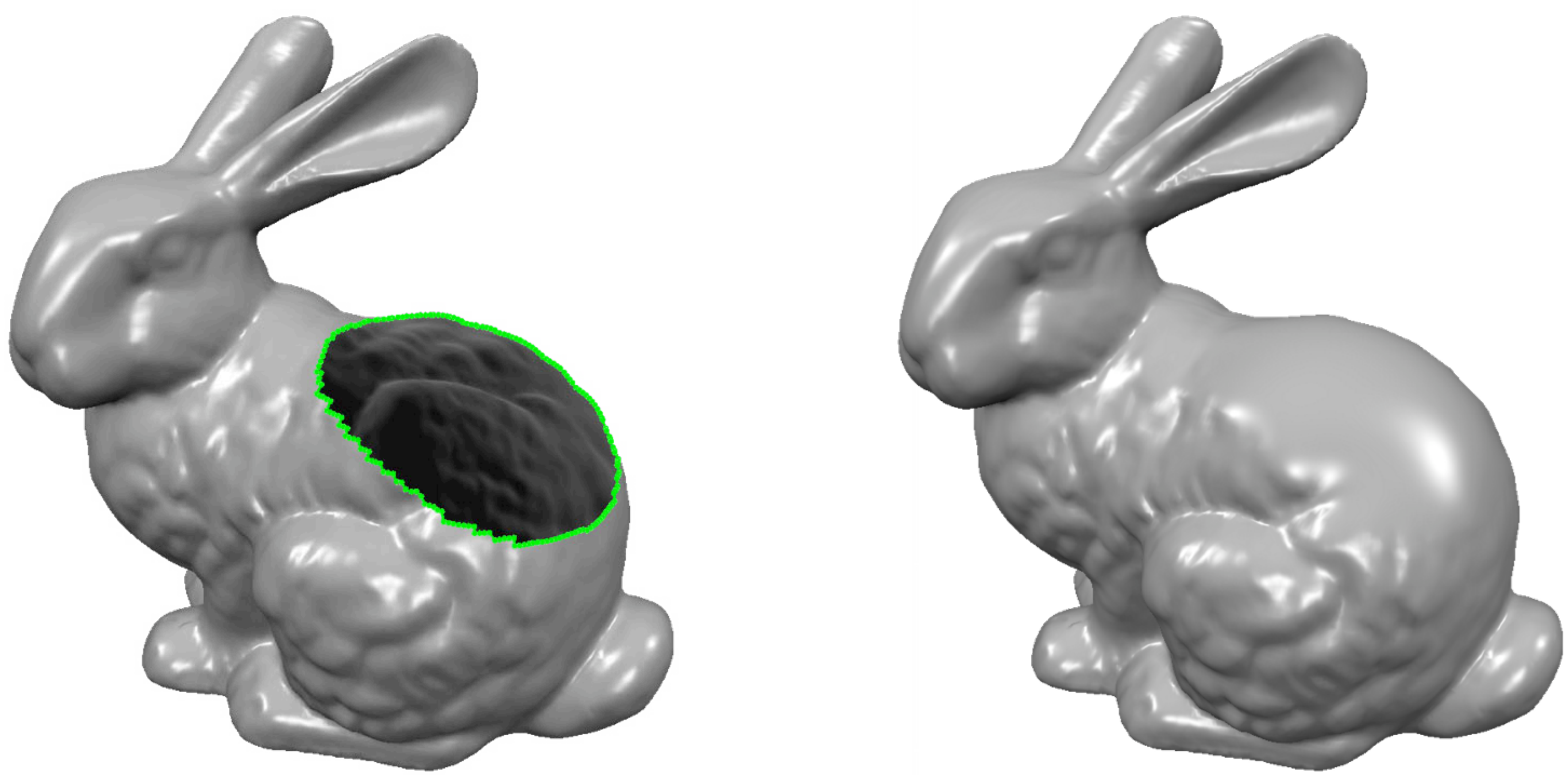
$\Delta \mathbf{x} \equiv 0$

$\Delta^2 \mathbf{x} \equiv 0$

$\Delta^3 \mathbf{x} \equiv 0$

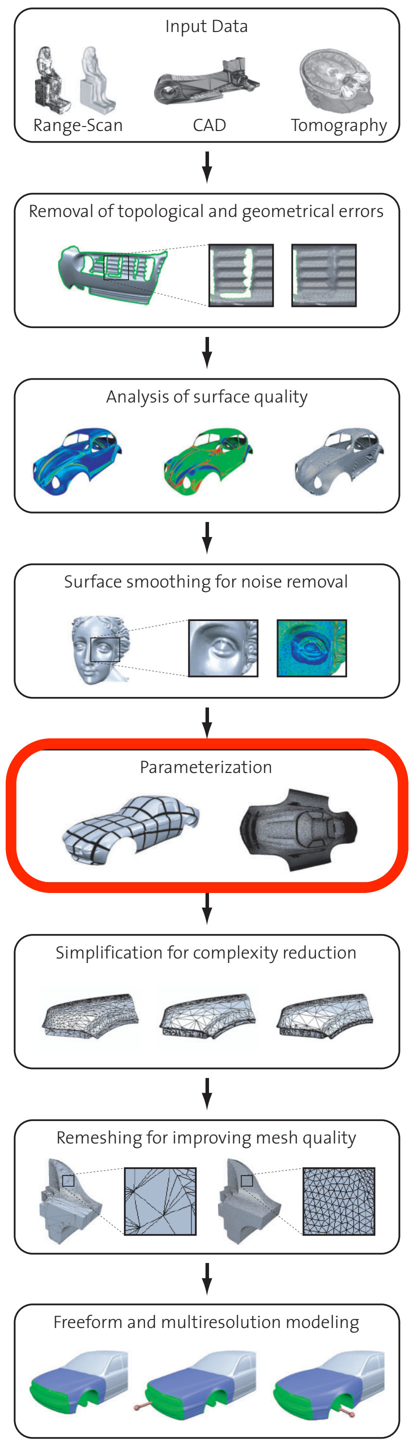
Explicit Hole Filling

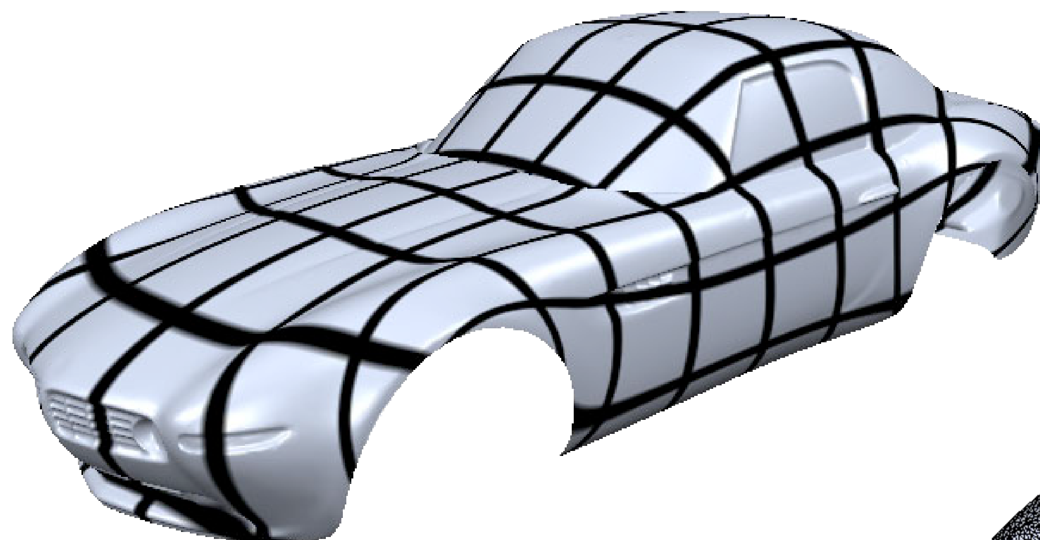




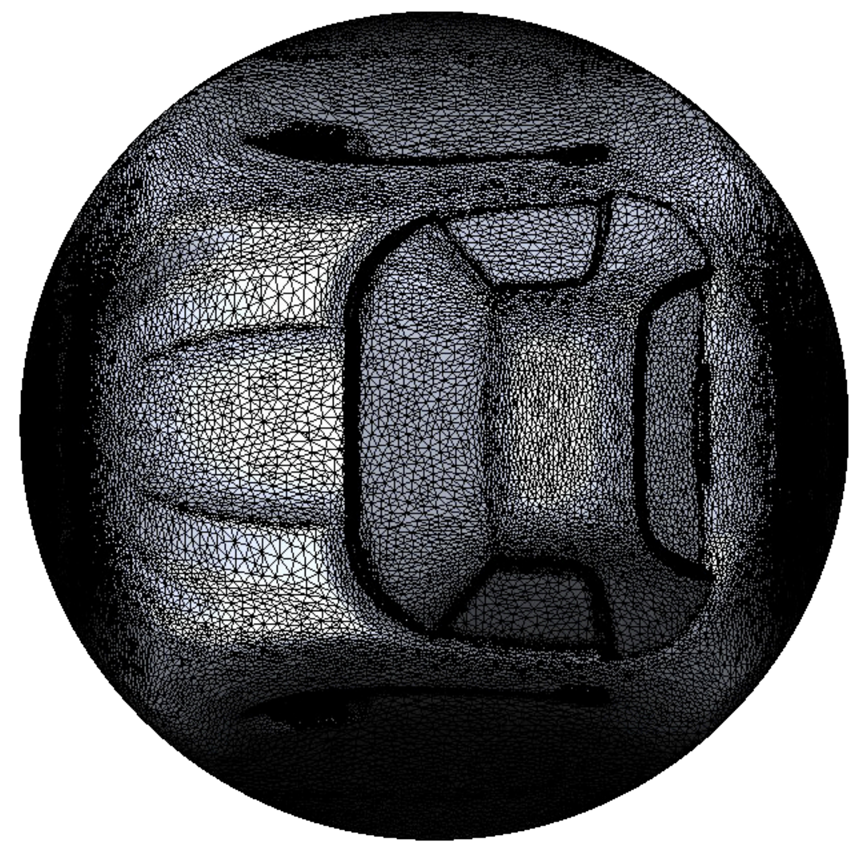
$$\Delta^2 \mathbf{x} = 0$$

Conformal Parameterization

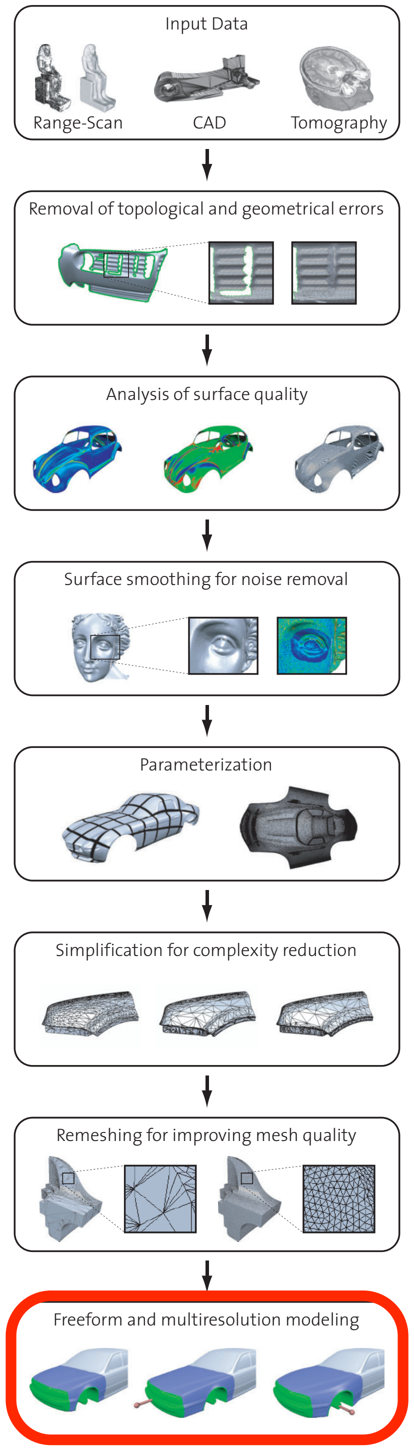


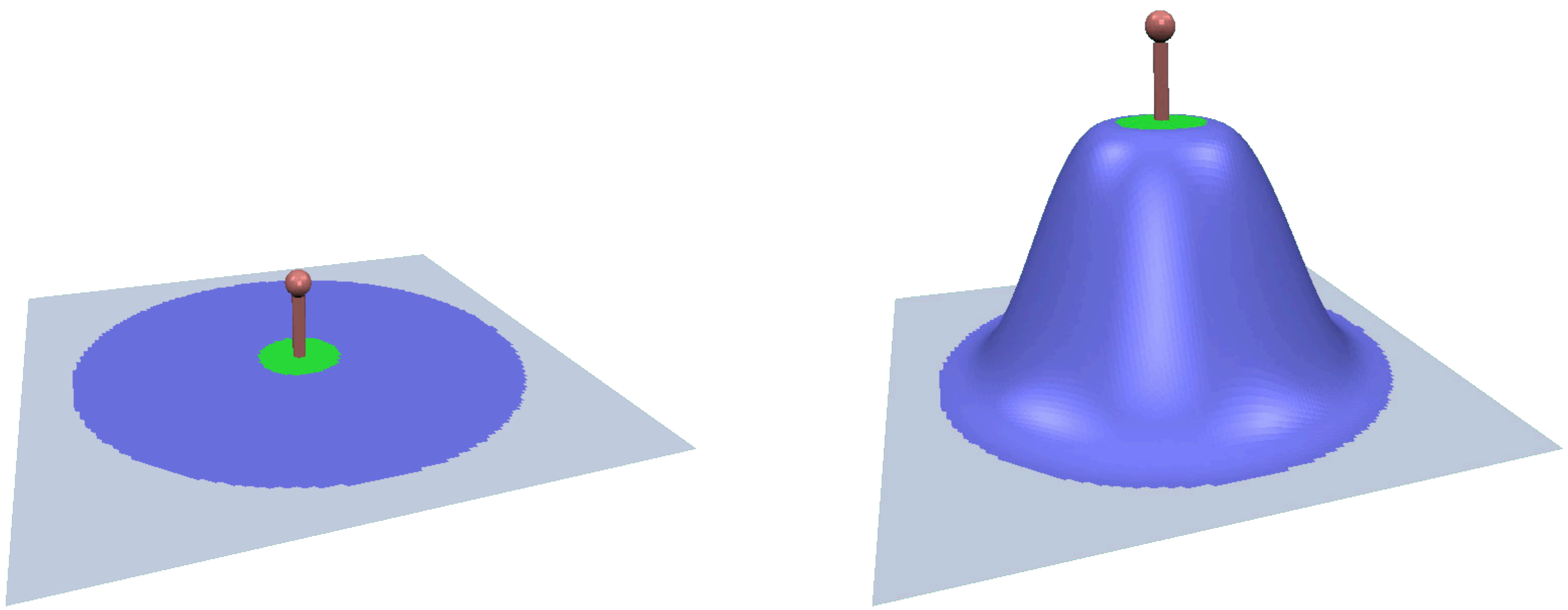


$$\Delta_{\mathcal{S}} \mathbf{u} = 0$$



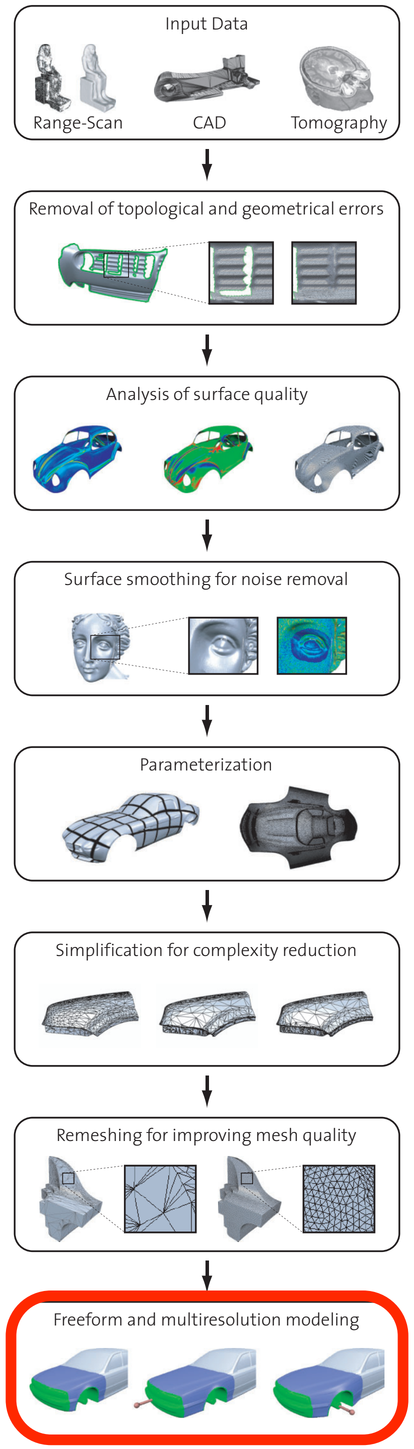
Variational Mesh Editing

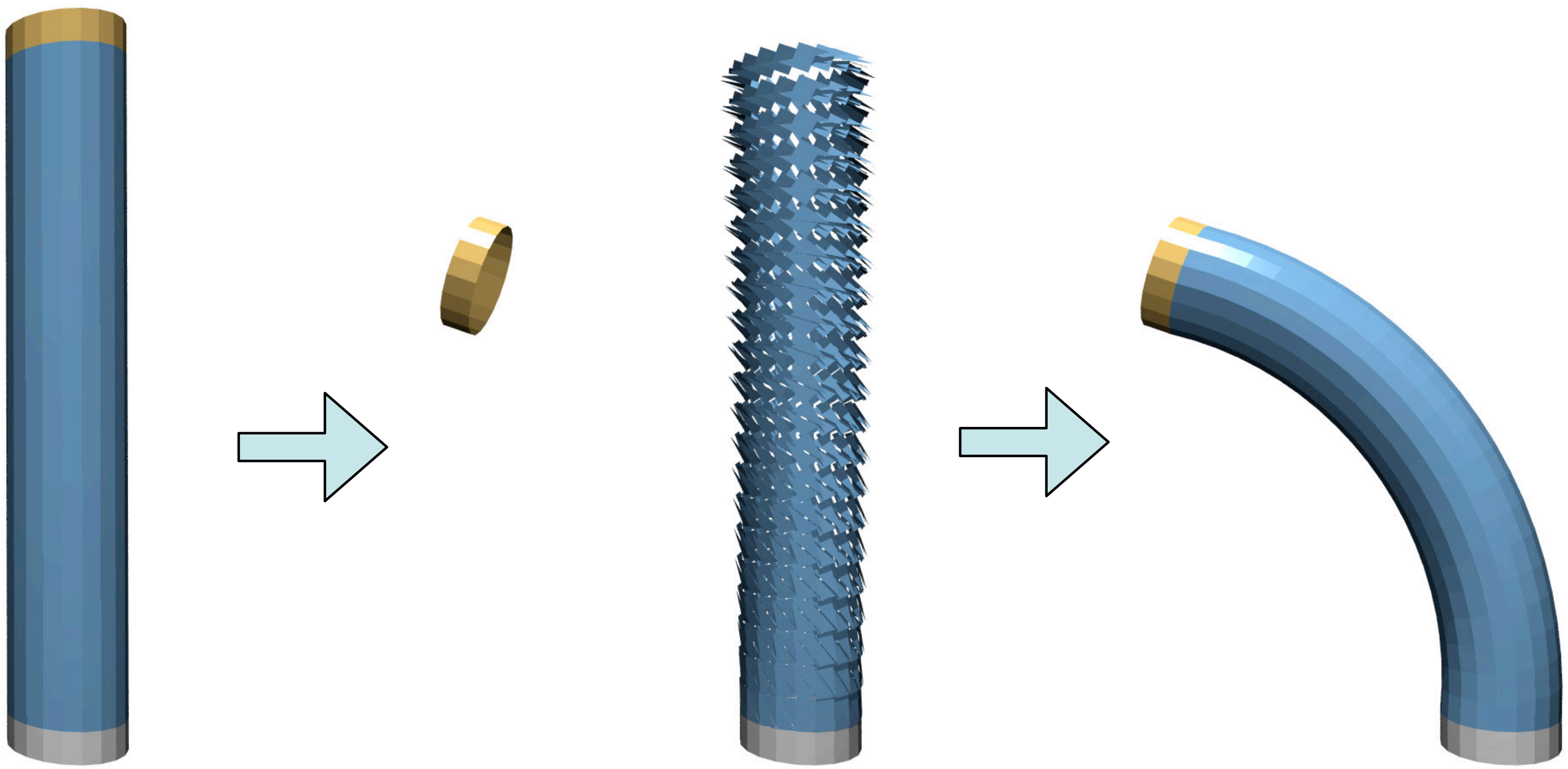




$$\Delta_S^k \mathbf{d} = 0$$

Gradient-Based Editing

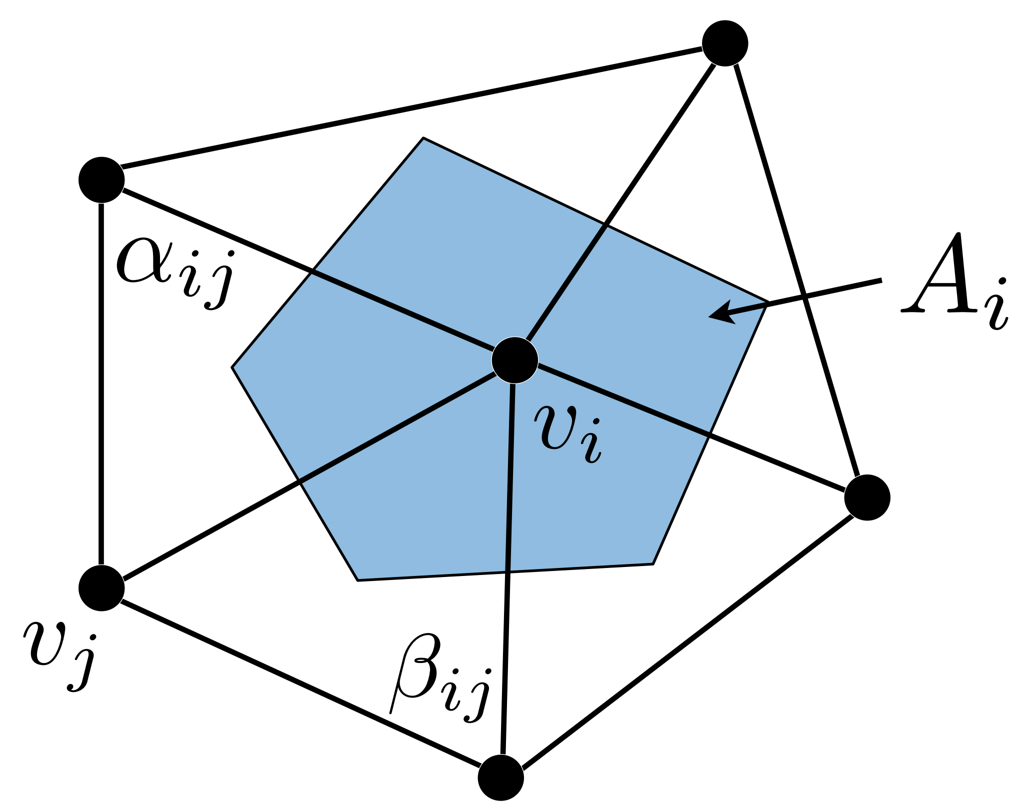




$$\Delta_{\mathcal{S}} \mathbf{p} = \operatorname{div}(\mathbf{g})$$

Laplace-Beltrami Discretization

$$\Delta_{\mathcal{S}} f(v) := \frac{2}{A(v)} \sum_{v_i \in \mathcal{N}_1(v)} (\cot\alpha_i + \cot\beta_i)\,(f(v_i) - f(v))$$



Laplace Matrix

$$\begin{pmatrix} \vdots \\ \Delta_{\mathcal{S}}^{k} f_i \\ \vdots \end{pmatrix} = (\mathbf{DM})^k \begin{pmatrix} \vdots \\ f_i \\ \vdots \end{pmatrix}$$

$$\mathbf{M}_{ij} = \begin{cases} \cot\alpha_{ij} + \cot\beta_{ij}, & i \neq j \;,\; j \in \mathcal{N}_1(v_i) \\ 0 & i \neq j \;,\; j \notin \mathcal{N}_1(v_i) \\ -\sum_{v_j \in \mathcal{N}_1(v_i)} (\cot\alpha_{ij} + \cot\beta_{ij}) & i = j \end{cases}$$

$$\mathbf{D} = \operatorname{diag}\left(\ldots, \frac{2}{A(v_i)}, \ldots\right)$$

Laplace Matrix

$$\begin{pmatrix} \vdots \\ \Delta_{\mathcal{S}}^k f_i \\ \vdots \end{pmatrix} = (\mathbf{DM})^k \begin{pmatrix} \vdots \\ f_i \\ \vdots \end{pmatrix}$$

- Degree of sparsity: $1 + 3\,(k^2 + k)$
  - k=1 ... 7
  - k=2 ... 19
  - k=3 ... 37

Laplace Matrix

$$\begin{pmatrix} \vdots \\ \Delta_{\mathcal{S}}^k f_i \\ \vdots \end{pmatrix} = (\mathbf{DM})^k \begin{pmatrix} \vdots \\ f_i \\ \vdots \end{pmatrix}$$

- $(\mathbf{DM})^k$ is not symmetric, but $\mathbf{M}(\mathbf{DM})^{k-1}$ is

➡ Instead of $(\mathbf{DM})^k\ \mathbf{x} = \mathbf{b}$

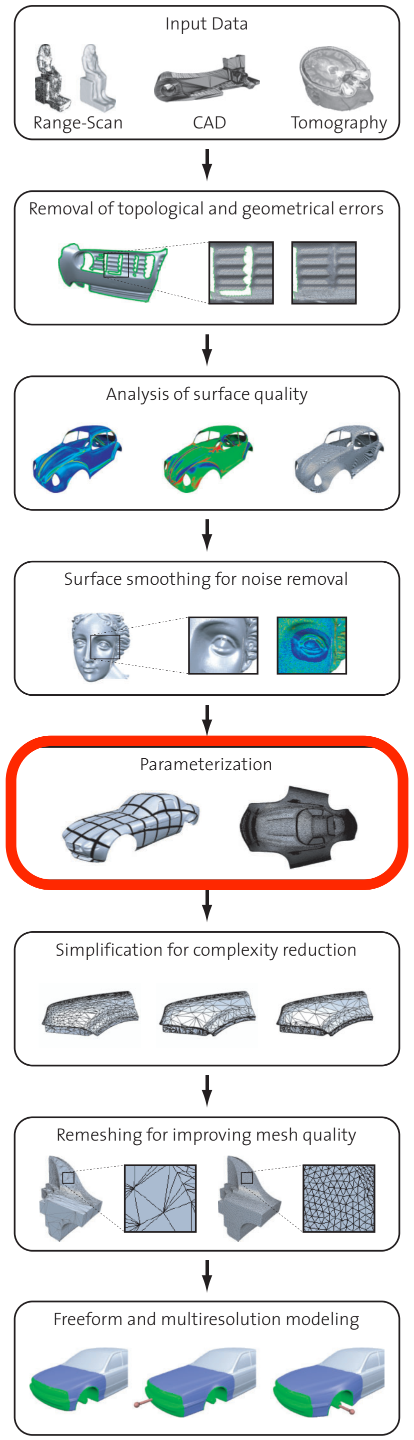
solve $\mathbf{M}(\mathbf{DM})^{k-1}\ \mathbf{x} = \mathbf{D}^{-1}\mathbf{b}$

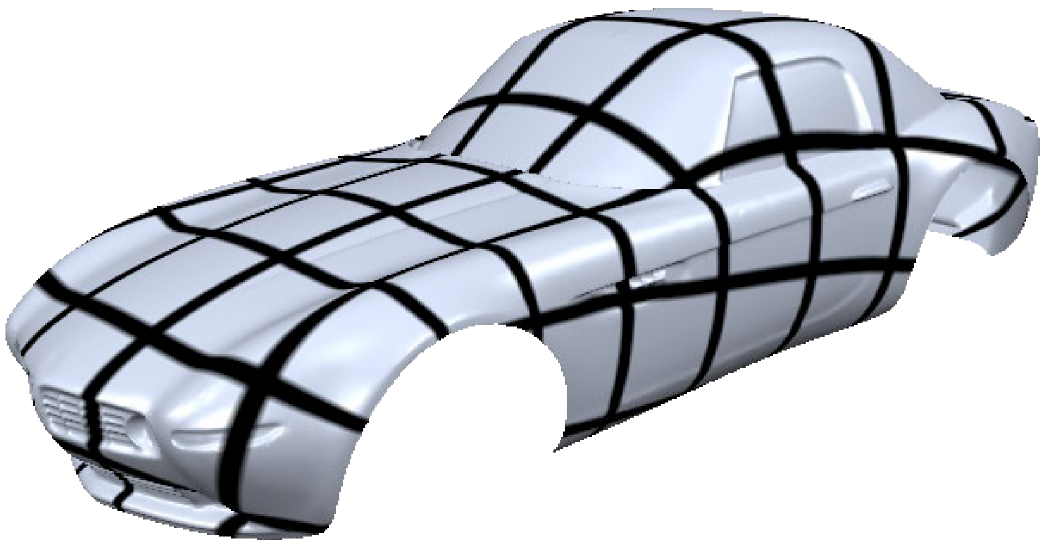
Laplace Matrix

$$\begin{pmatrix} \vdots \\ \Delta_{\mathcal{S}}^k f_i \\ \vdots \end{pmatrix} = (\mathbf{DM})^k \begin{pmatrix} \vdots \\ f_i \\ \vdots \end{pmatrix}$$

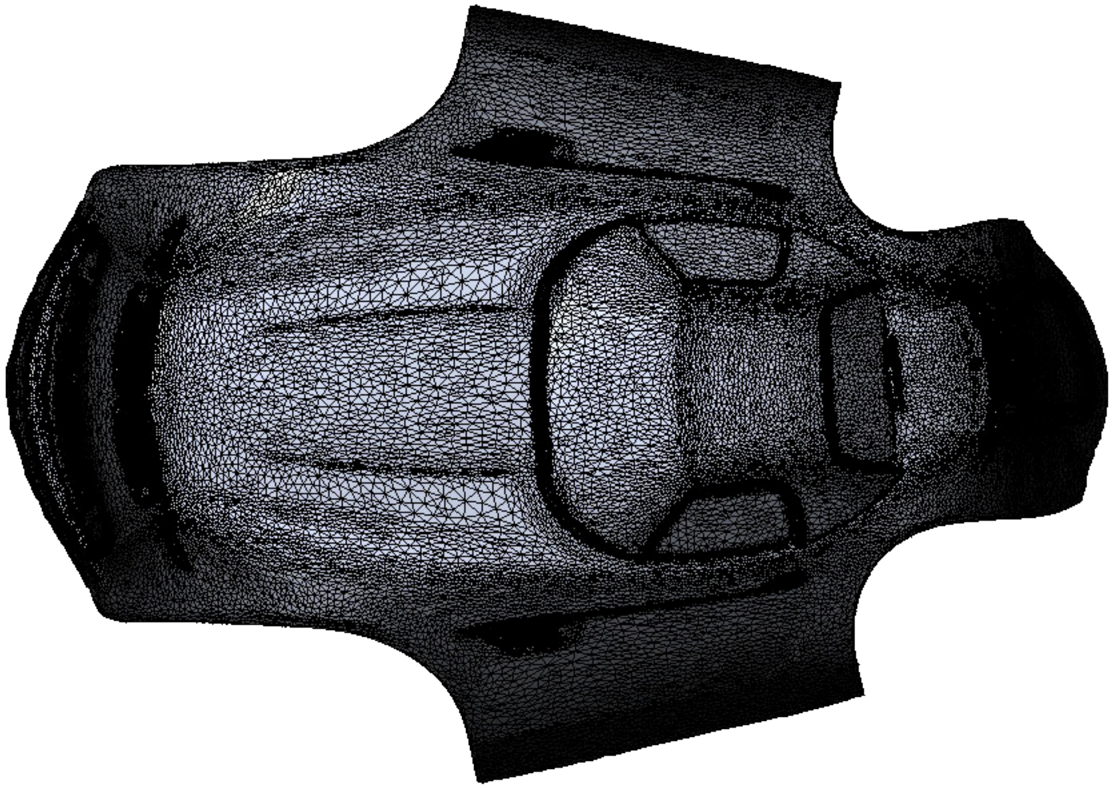
- Positive definiteness
  - Can be derived by variational calculus
  - Energy minimization subject to constraints

Least Squares Conformal Maps

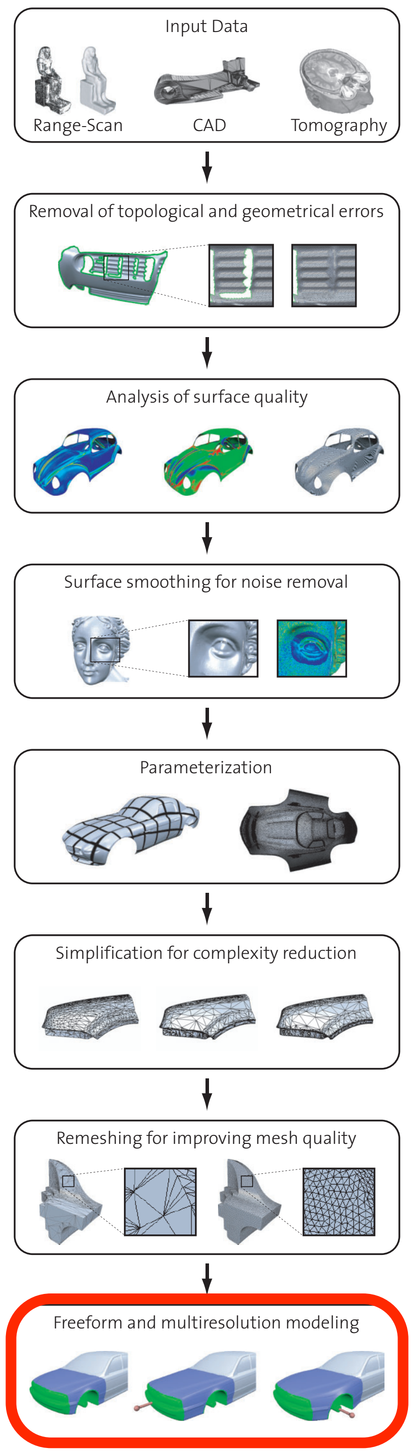




$$\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b}$$



Non-Linear Problems



Non-linear minimization (Newton)

$$\mathbf{H}(\mathbf{x})\,\mathbf{h} = -\nabla\mathbf{f}(\mathbf{x})$$

Non-linear least squares (Gauss-Newton)

$$\mathbf{J}(\mathbf{x})^T\,\mathbf{J}(\mathbf{x})\,\mathbf{h} = -\mathbf{J}(\mathbf{x})^T\,\mathbf{f}(\mathbf{x})$$

Overview

- Application scenarios
- **Linear system solvers**
- Benchmarks

Dense Direct Solvers

- Symmetric positive definite (*spd*)
  - Cholesky factorization ($\mathbf{A}=\mathbf{L}\mathbf{L}^T$)
  - Solve systems by back-substitution
  - Numerically stable
- Complexity
  - Factorization $O(n^3)$
  - Back-substitution $O(n^2)$

Iterative Solvers

- Symmetric, positive definite, sparse
  - Conjugate gradients
  - Krylov spaces $\mathbf{K}_i = \{ \mathbf{b}, \mathbf{Ab}, \ldots, \mathbf{A}^{i-1}\mathbf{b} \}$
  - Robust, monotone convergence
  - Exact solution after n iterations

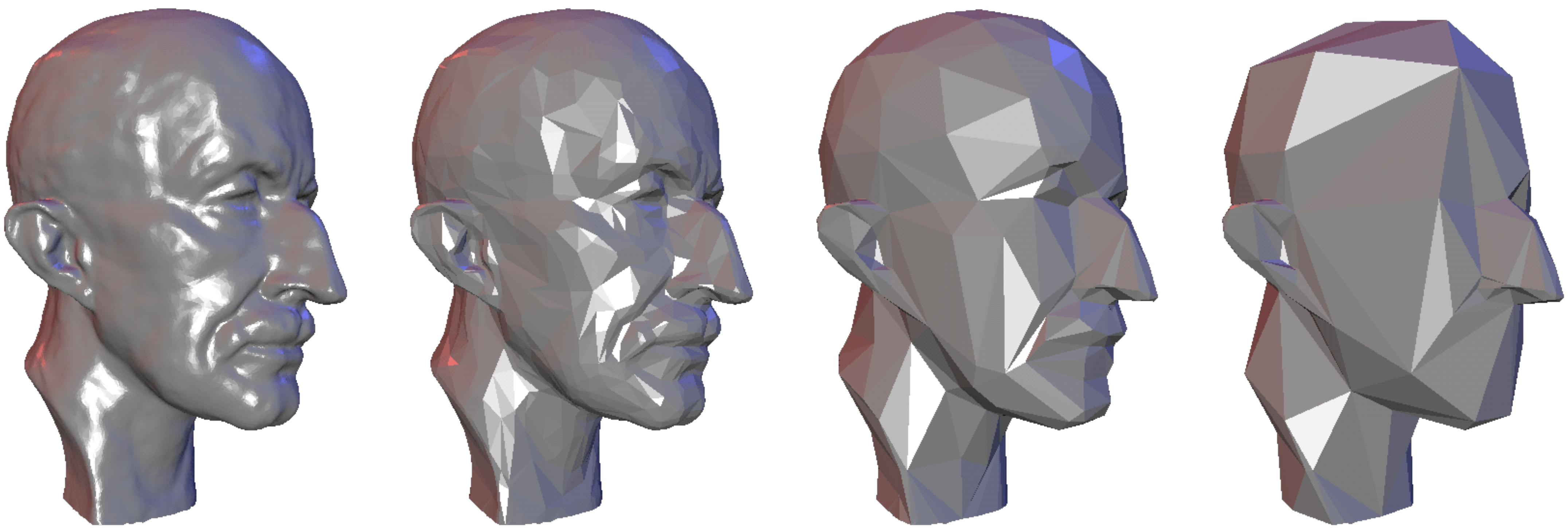
- Complexity
  - Each iteration is O(n) (<u>sparse</u>!)
  - Total complexity $O(n^2)$

Iterative Solvers

- Numerical convergence rate
  - Depends on matrix condition
  - Preconditioning is mandatory ($\mathbf{A}' = \mathbf{P}\mathbf{A}\mathbf{P}^T$)
  - Problematic for large systems ...
- Iterative solvers are “smoothers”
  - Rapid elimination of high frequency errors
  - Impractically slow convergence for low frequencies

Multigrid Solvers

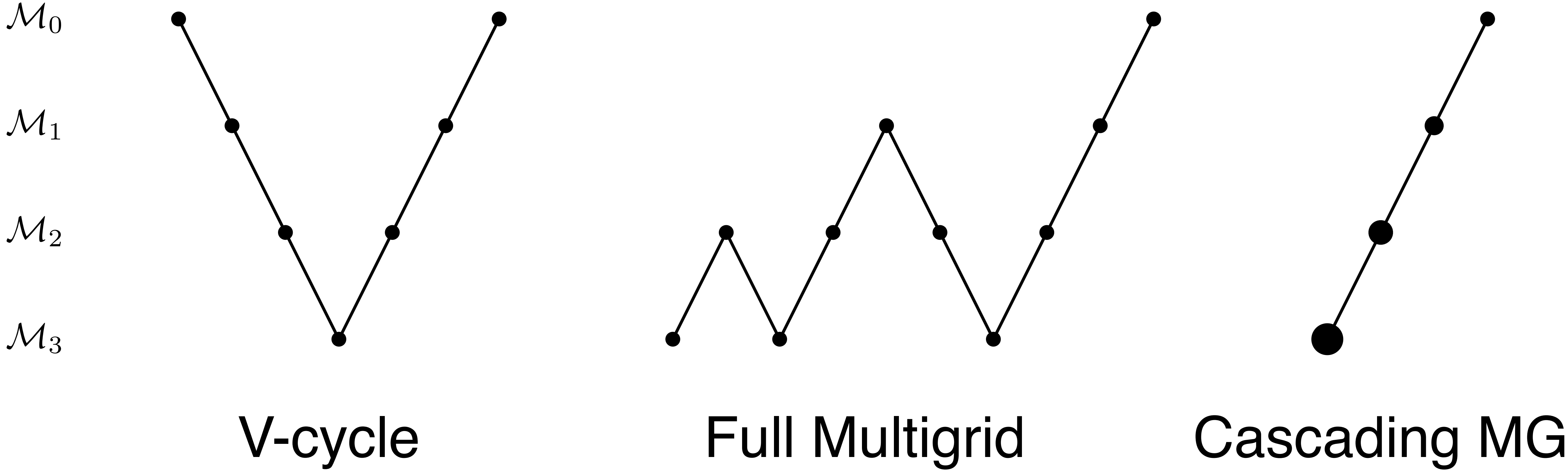
- Build a hierarchy of meshes
  - Mesh decimation
  - O(log n) levels



Multigrid Solvers

- Apply some pre-smoothing steps on finest level
  - Removes highest error frequencies
- Remaining low frequency error (**r**=**b**-**Ax**)
  - Corresponds to high frequencies on coarser levels
  - Iterate / solve residual system (**Ae**=**r**) on coarse level
- Propagate solution to finer level
  - Followed by post-smoothing steps
- Total O(n) complexity!

Multigrid Solvers



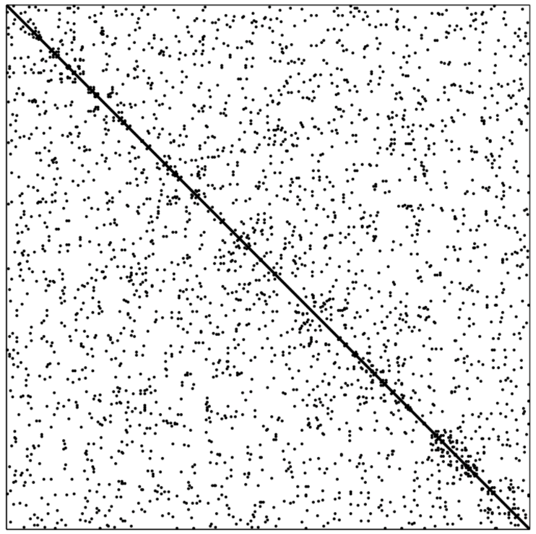
Multigrid Solvers

- MG can be quite tricky:
  - How to build an irregular hierarchy ?
  - How many levels ?
  - Special MG pre-conditioners
  - Restriction of system
  - Prolongation of coarse solution

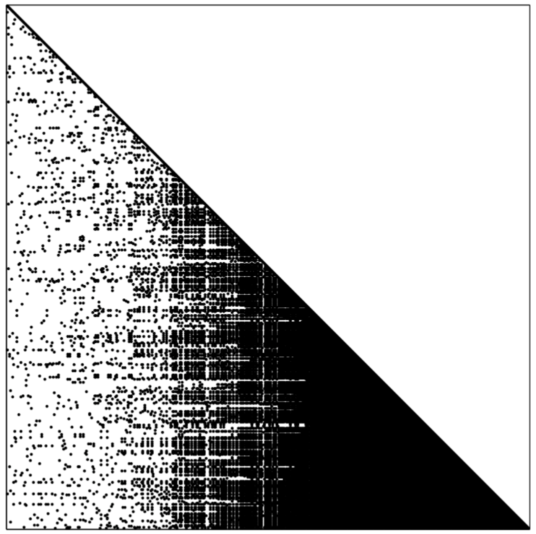
- [Aksoylu et al. 2003], [Shi et al. 2006]

Direct Sparse Solvers

- Dense solvers do not exploit sparsity
  - Matrix factors are dense



$A=LL^T$

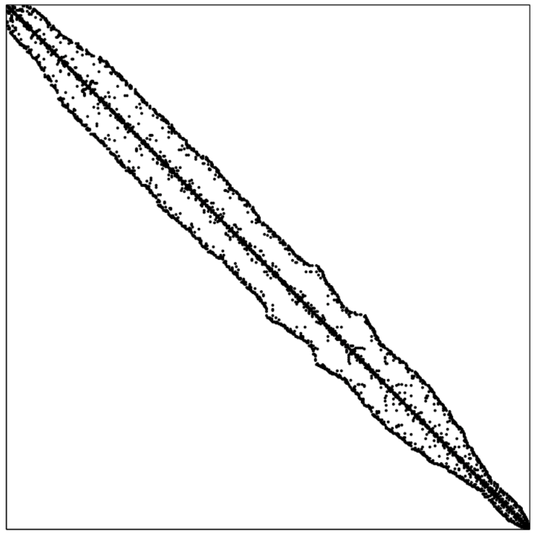
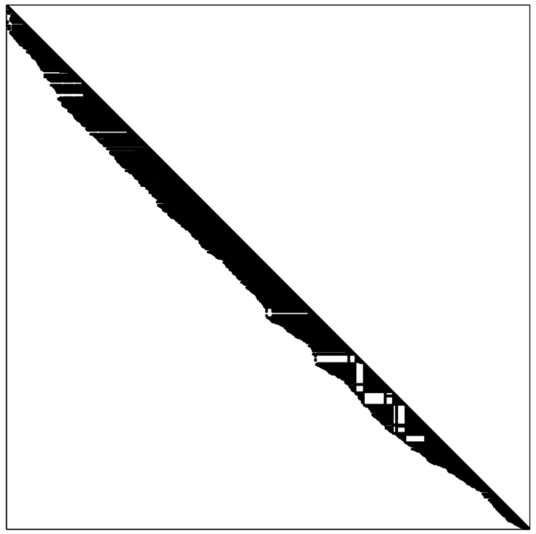


L

Direct Sparse Solvers

- Dense solvers do not exploit sparsity
  - Matrix factors are dense

- Band-limitation can be exploited
  - Bandwidth of factors is that of **A**
  - More precisely: envelope is preserved

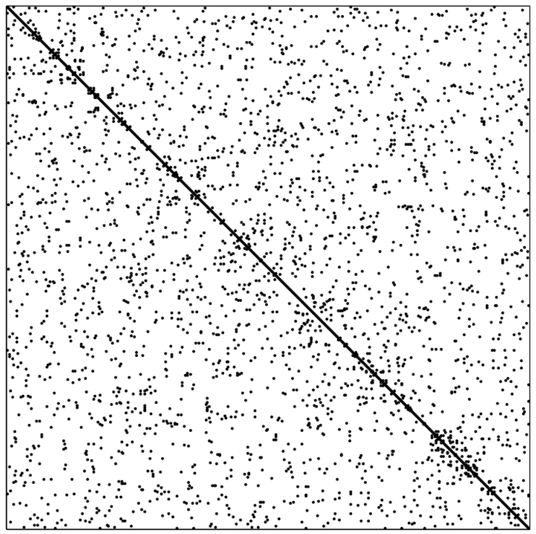
Direct Sparse Solvers

- Dense solvers do not exploit sparsity
  - Matrix factors are dense
- Band-limitation can be exploited
  - Bandwidth of factors is that of **A**
  - More precisely: envelope is preserved
- Complexity
  - Factorization $O(nb^2)$
  - Back-substitution $O(nb)$

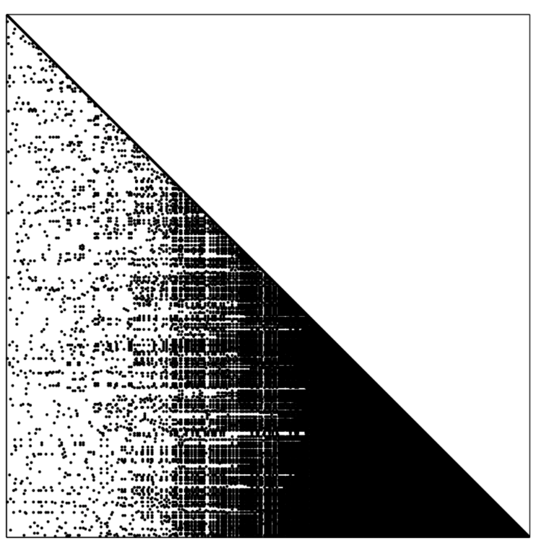
Matrix Re-Ordering

Natural

LL^T



L



36k NZ

Matrix Re-Ordering

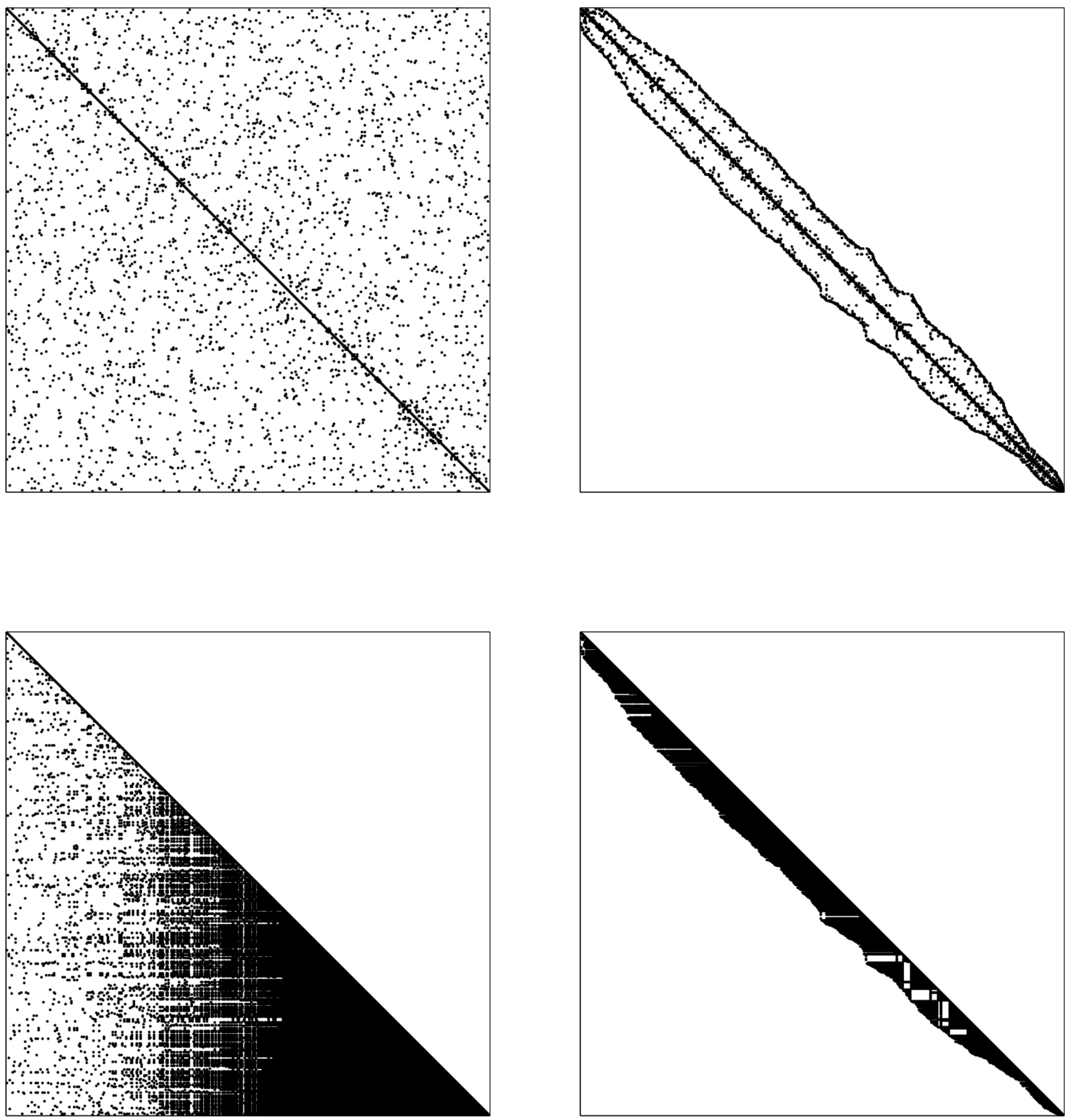
- Find symmetric permutation **A**' = $\mathbf{P}^T\mathbf{A}\mathbf{P}$
- ... which minimizes the band-width:
  - ➡ Cuthill-McKee algorithm

Matrix Re-Ordering

Natural RCMK

LL^T

L



36k NZ 14k NZ

Matrix Re-Ordering

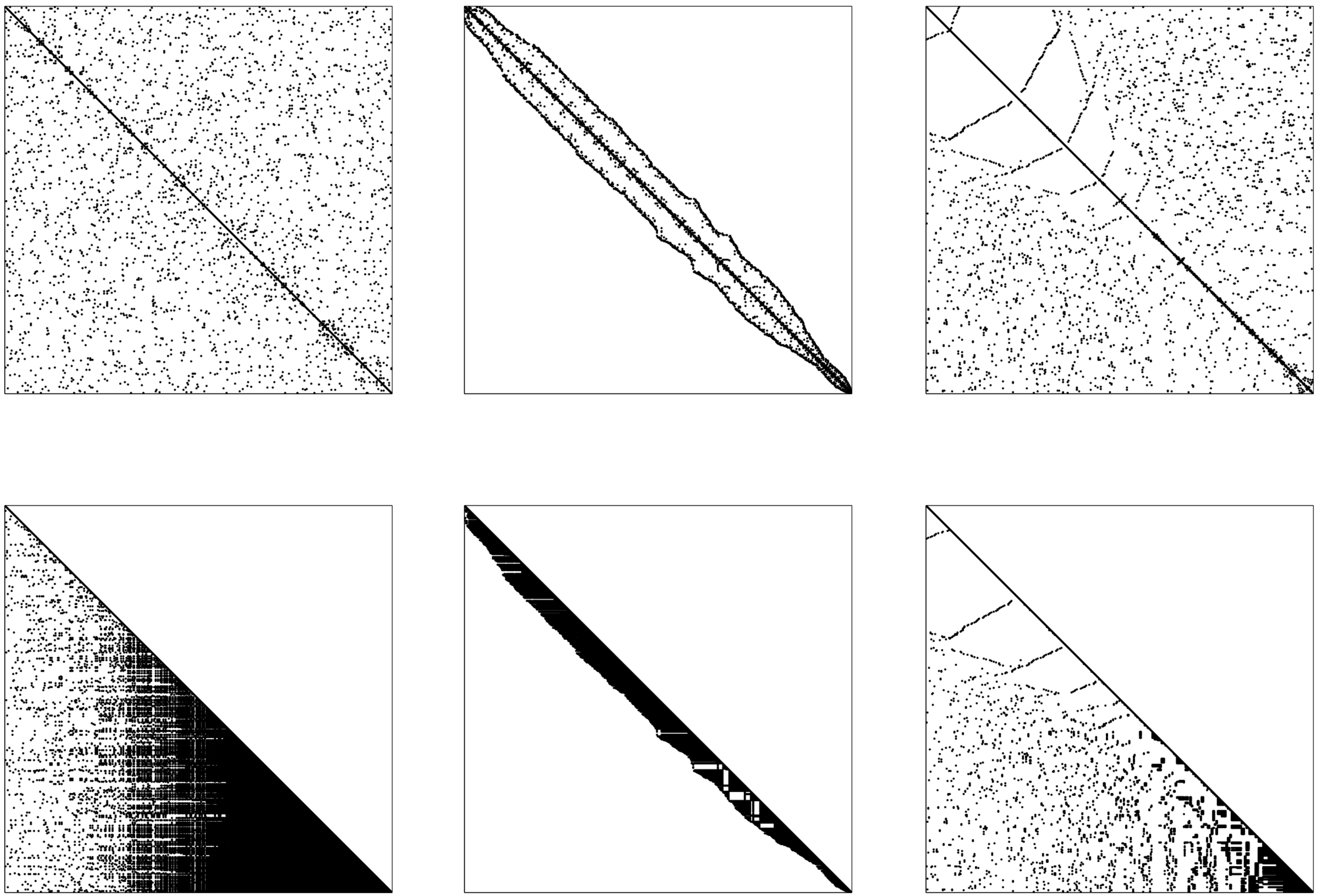
- Find symmetric permutation $\mathbf{A}' = \mathbf{P}^T \mathbf{A} \mathbf{P}$
- ... which minimizes the band-width:
  ➡ Cuthill-McKee algorithm
- ... which minimizes the envelope fill-in of $\mathbf{L}$:
  ➡ Minimum Degree algorithm

Matrix Re-Ordering

Natural RCMK MD

LL^T

L



36k NZ 14k NZ 6.2k NZ

Matrix Re-Ordering

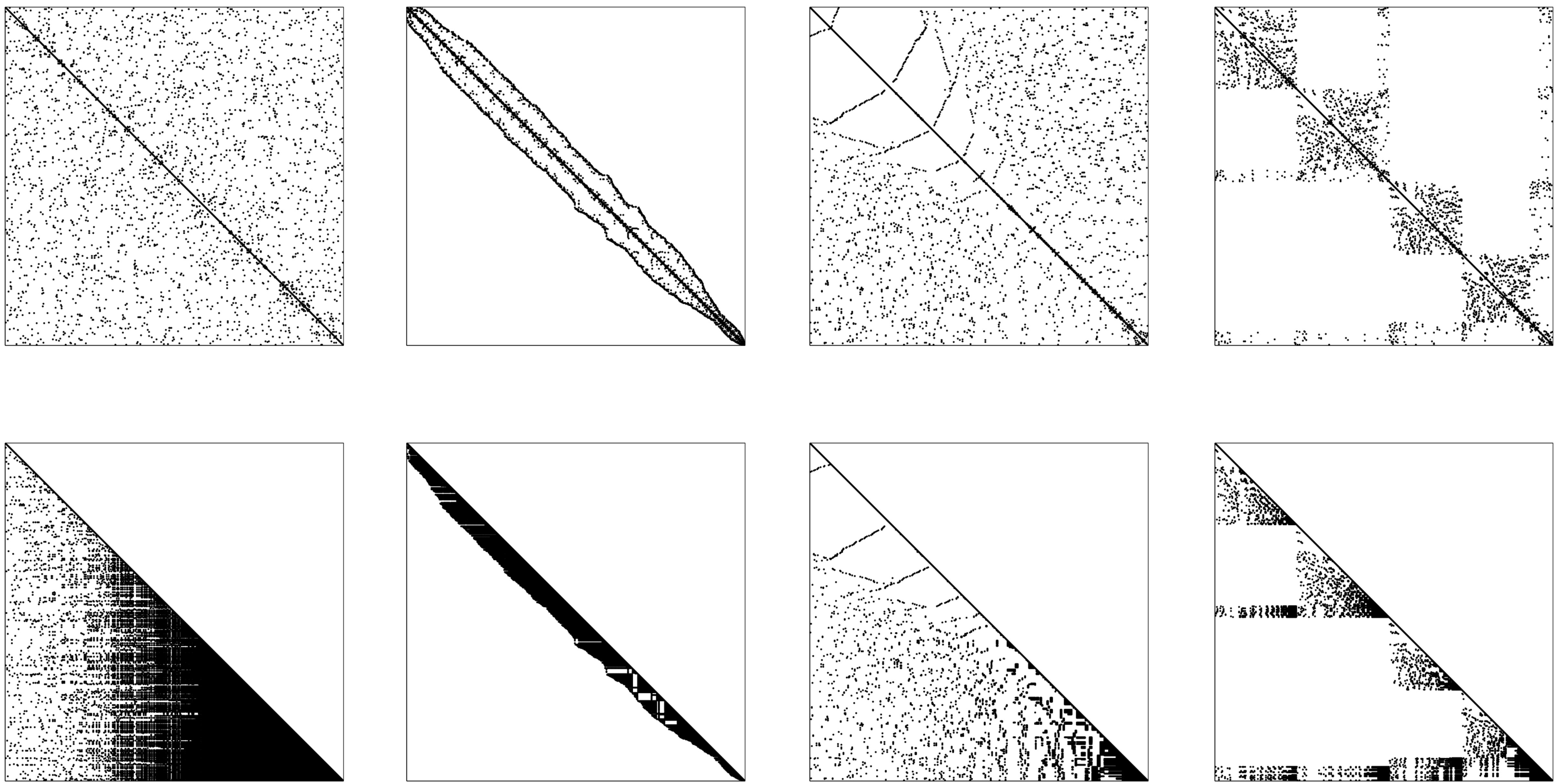
- Find symmetric permutation $\mathbf{A}' = \mathbf{P}^T\mathbf{AP}$
- ... which minimizes the band-width:
  ➡ Cuthill-McKee algorithm
- ... which minimizes the envelope fill-in of **L**:
  ➡ Minimum Degree algorithm
- ... based on recursive graph partitioning:
  ➡ METIS algorithm

Matrix Re-Ordering

Natural RCMK MD Metis

LL^T

L



36k NZ 14k NZ 6.2k NZ 7.1k NZ

Sparse Cholesky Factorization

- Non-zero structure of **L** can be predicted from the non-zero structure of **A**
  - Build a <u>static</u> data structure in advance
  - *Symbolic factorization*

- Compute numerical entries of **L** based on this data structure
  - Better memory coherence
  - *Numerical factorization*

Sparse Cholesky Solver

1. Matrix re-ordering $\tilde{\mathbf{A}} = \mathbf{P}^T \mathbf{A} \mathbf{P}$
2. Symbolic factorization $\mathbf{L}$
3. Numerical factorization $\tilde{\mathbf{A}} = \mathbf{L}\mathbf{L}^T$
4. Solve system $\mathbf{y} = \mathbf{L}^{-1} \mathbf{P}^T \mathbf{b}, \quad \mathbf{x} = \mathbf{P}\mathbf{L}^{-T} \mathbf{y}$

Sparse Cholesky Solver

Only right hand side changes

1. Matrix re-ordering $\tilde{\mathbf{A}} = \mathbf{P}^T \mathbf{A} \mathbf{P}$
2. Symbolic factorization $\mathbf{L}$
3. Numerical factorization $\tilde{\mathbf{A}} = \mathbf{L}\mathbf{L}^T$
4. Solve system $\mathbf{y} = \mathbf{L}^{-1}\mathbf{P}^T\mathbf{b}, \quad \mathbf{x} = \mathbf{P}\mathbf{L}^{-T}\mathbf{y}$

Sparse Cholesky Solver

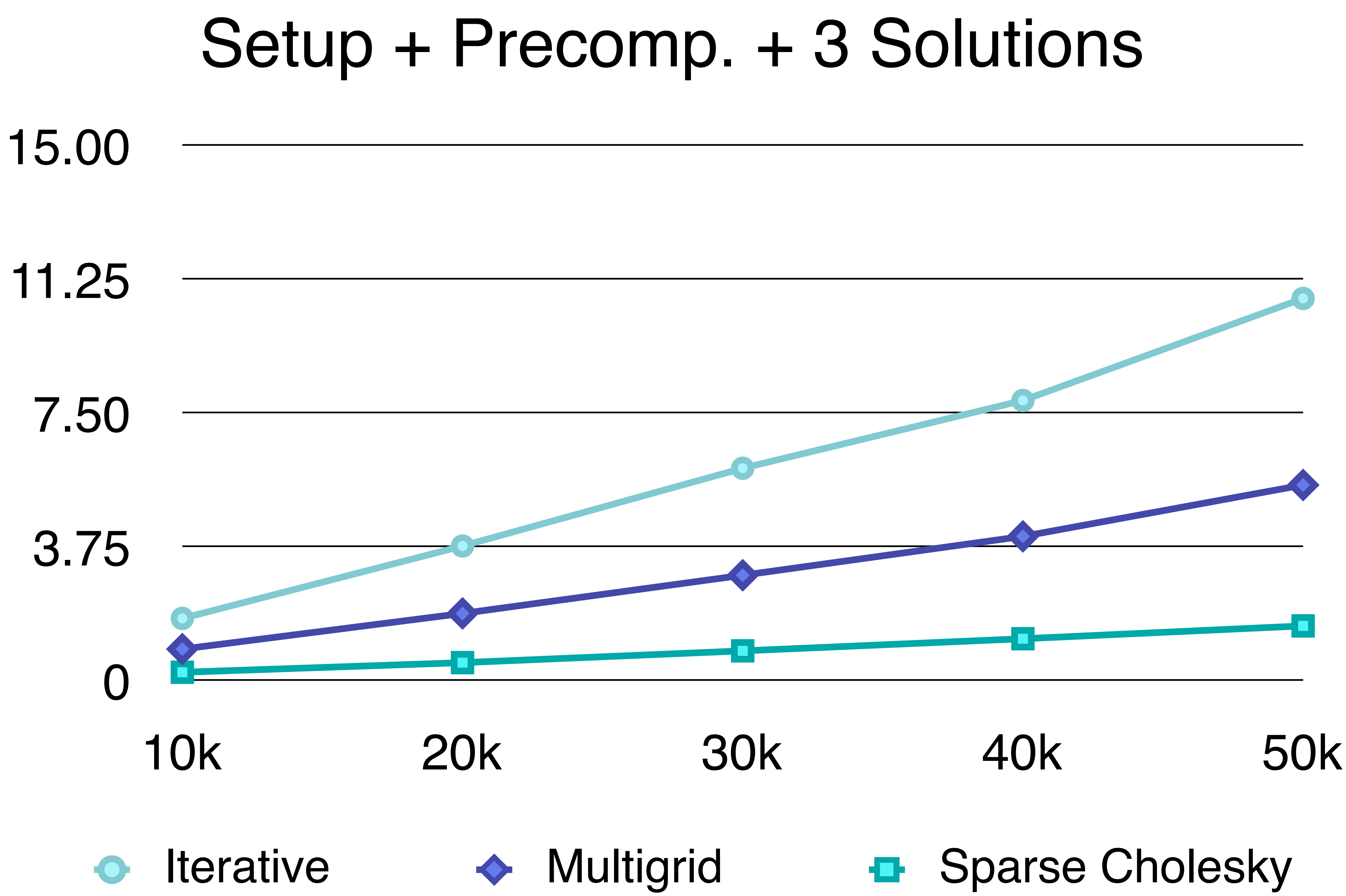
Matrix values change

1. Matrix re-ordering $\tilde{\mathbf{A}} = \mathbf{P}^T \mathbf{A} \mathbf{P}$
2. Symbolic factorization $\mathbf{L}$
3. Numerical factorization $\tilde{\mathbf{A}} = \mathbf{L}\mathbf{L}^T$
4. Solve system $\mathbf{y} = \mathbf{L}^{-1} \mathbf{P}^T \mathbf{b}, \quad \mathbf{x} = \mathbf{P} \mathbf{L}^{-T} \mathbf{y}$

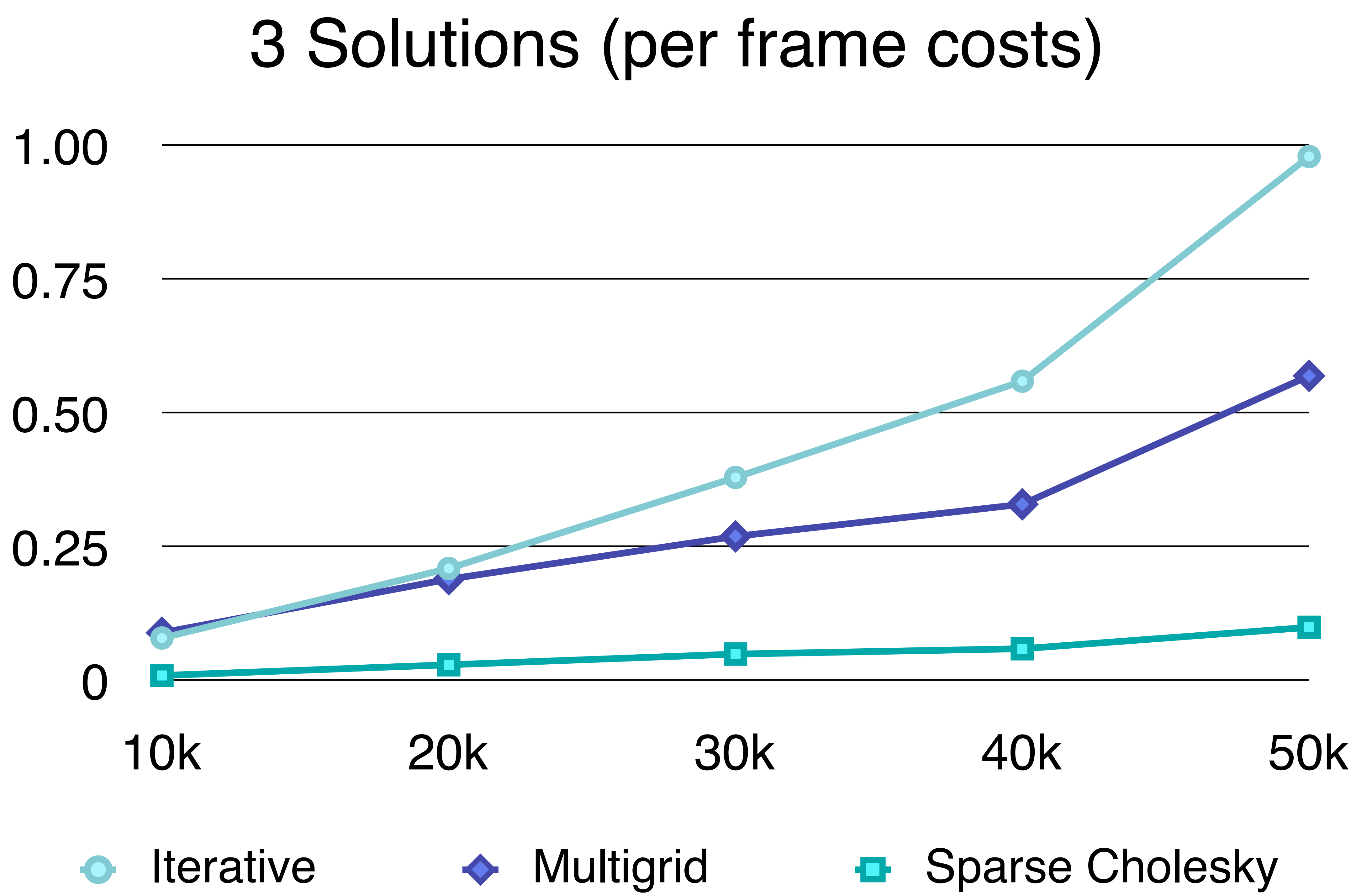
Overview

- Application scenarios
- Linear system solvers
- **Benchmarks**

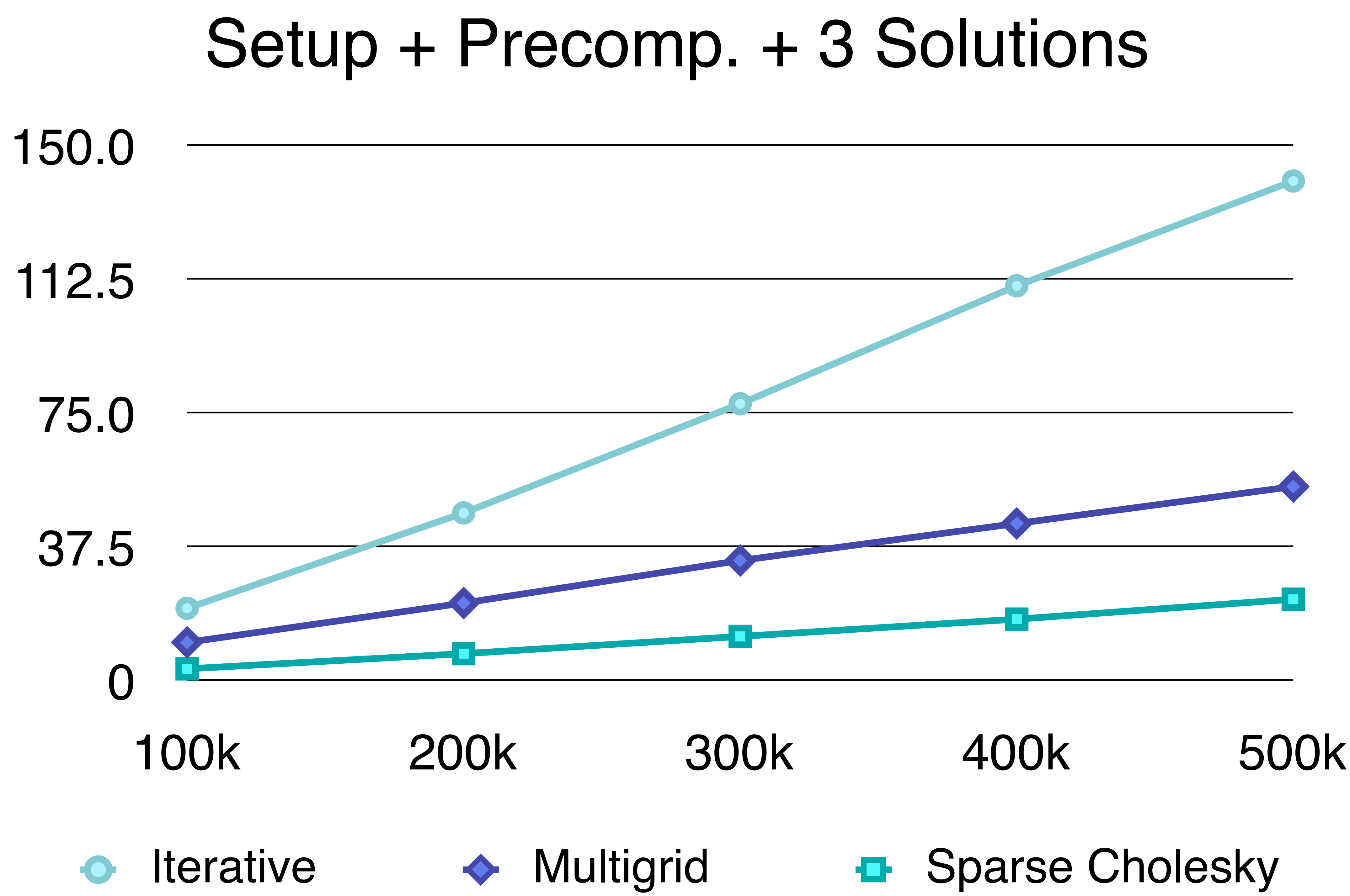
Small Laplace Systems



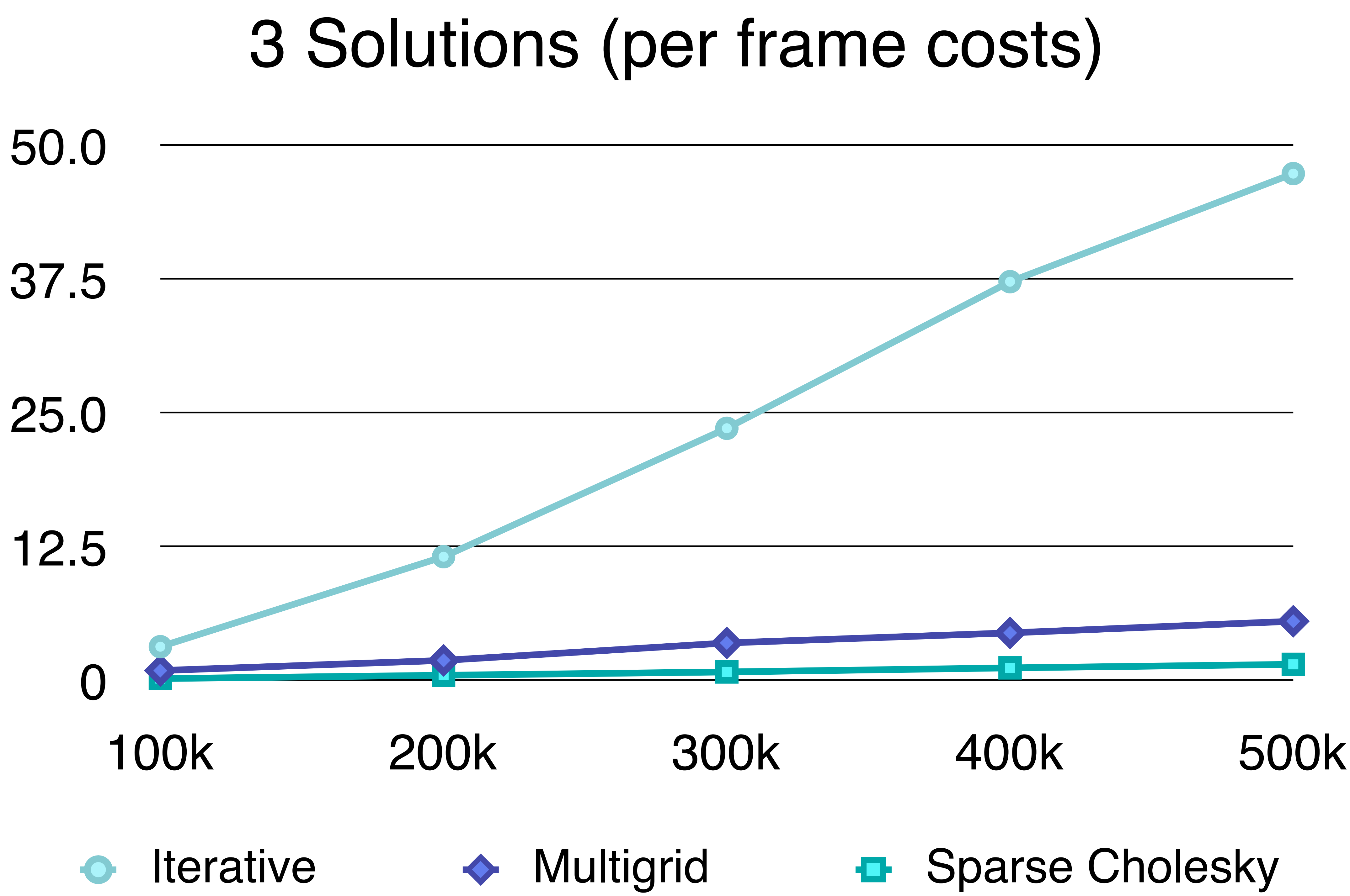
Small Laplace Systems



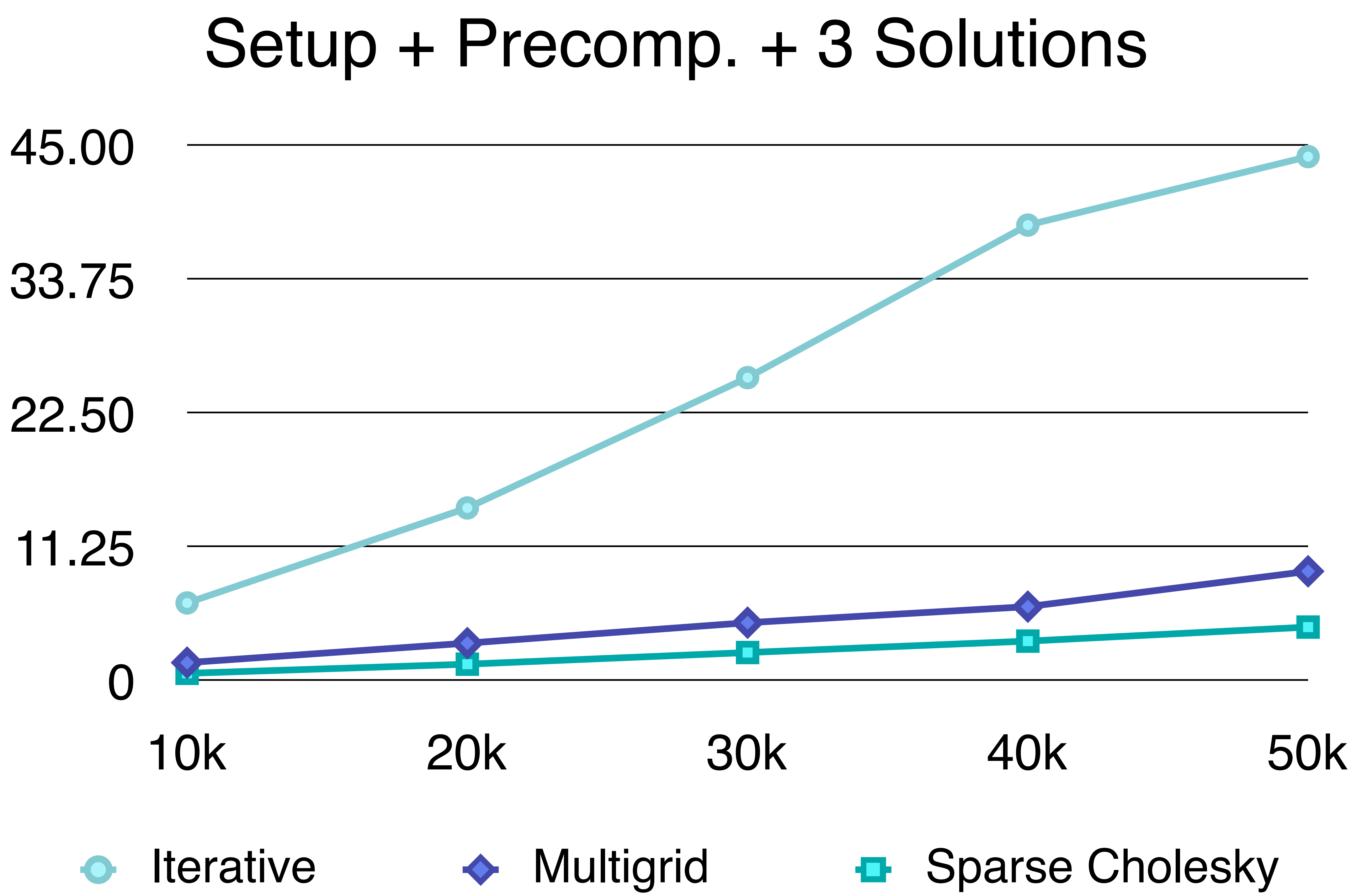
Large Laplace Systems



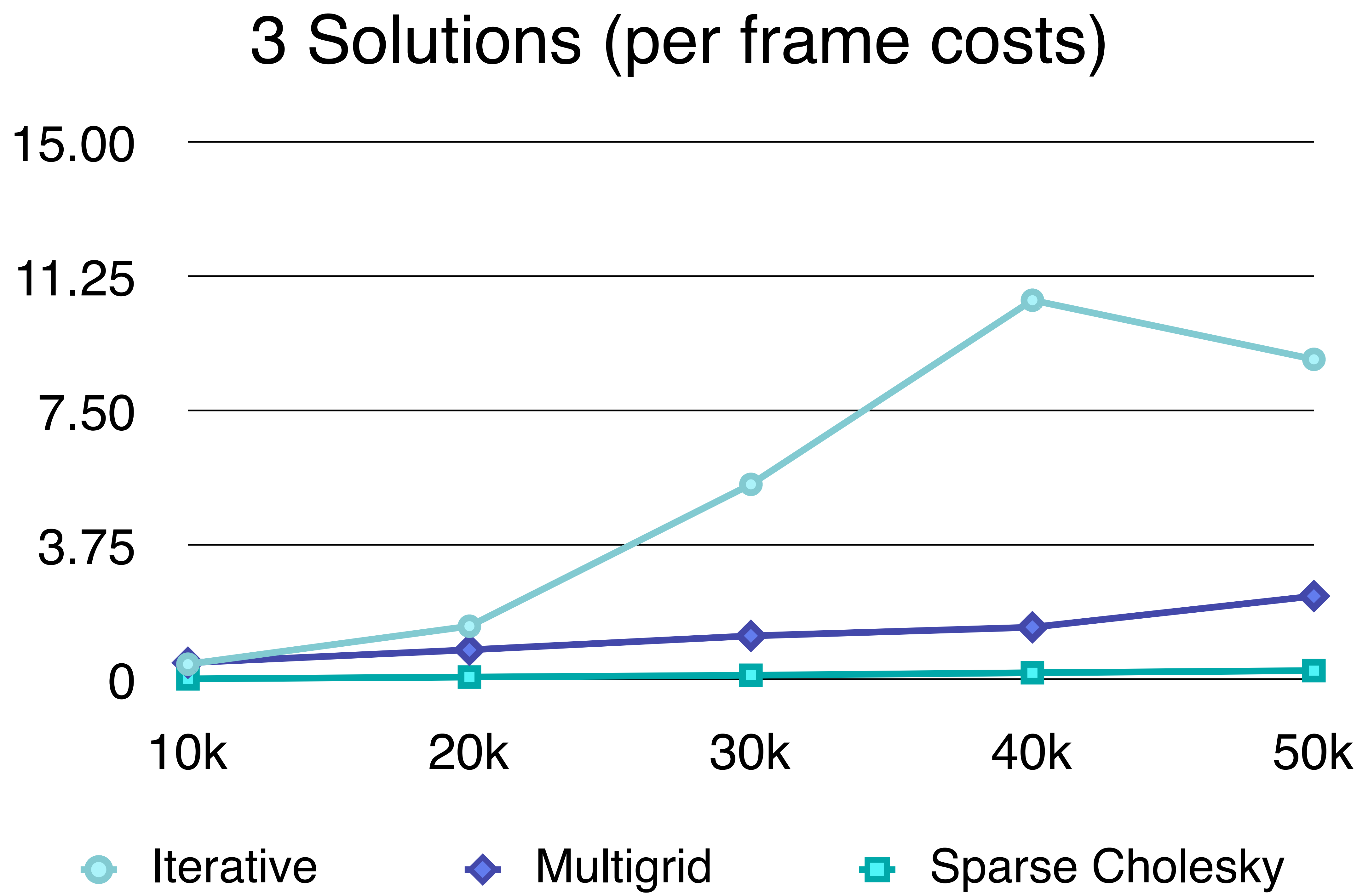
Large Laplace Systems



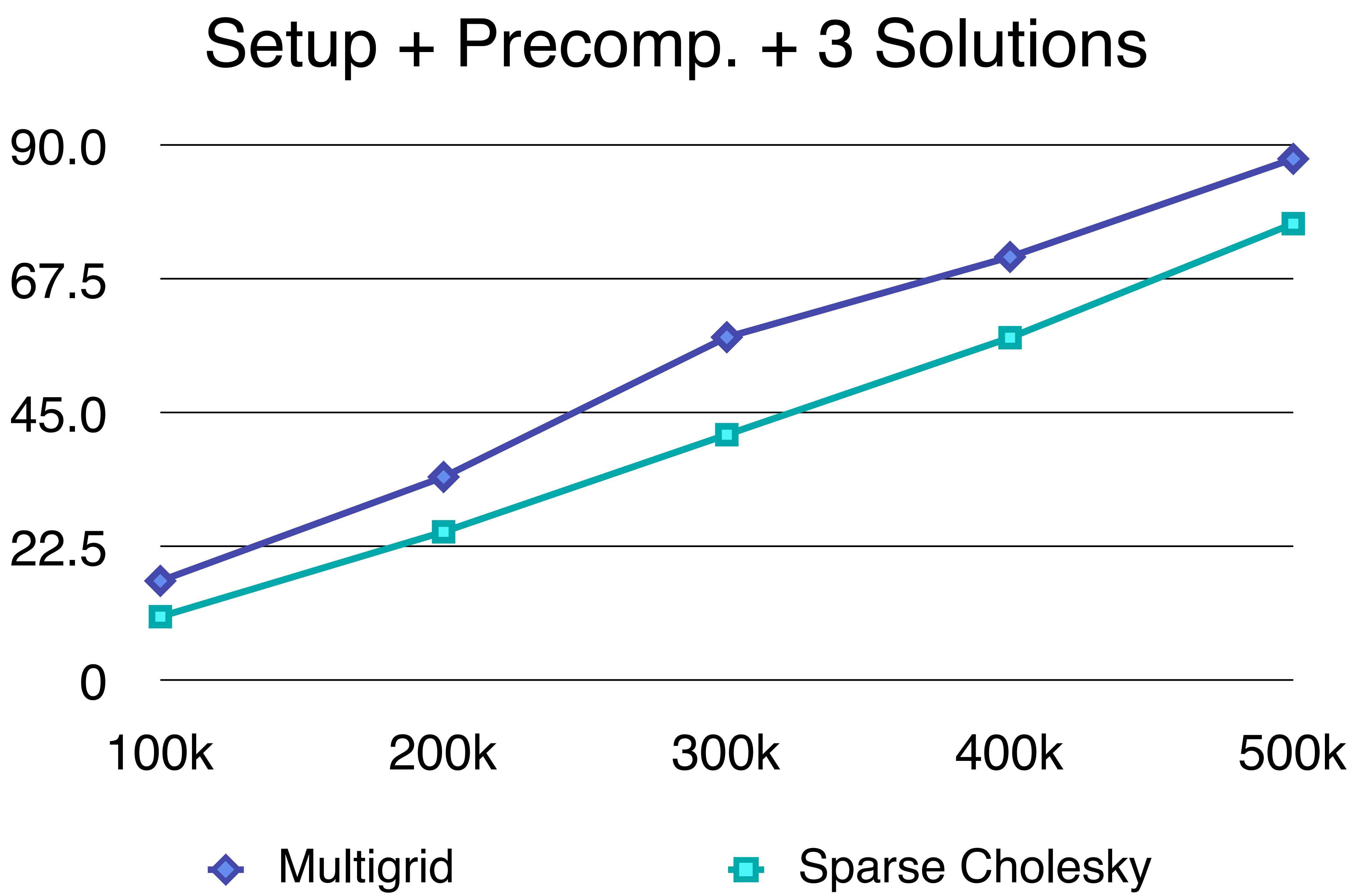
Small Bi-Laplace Systems



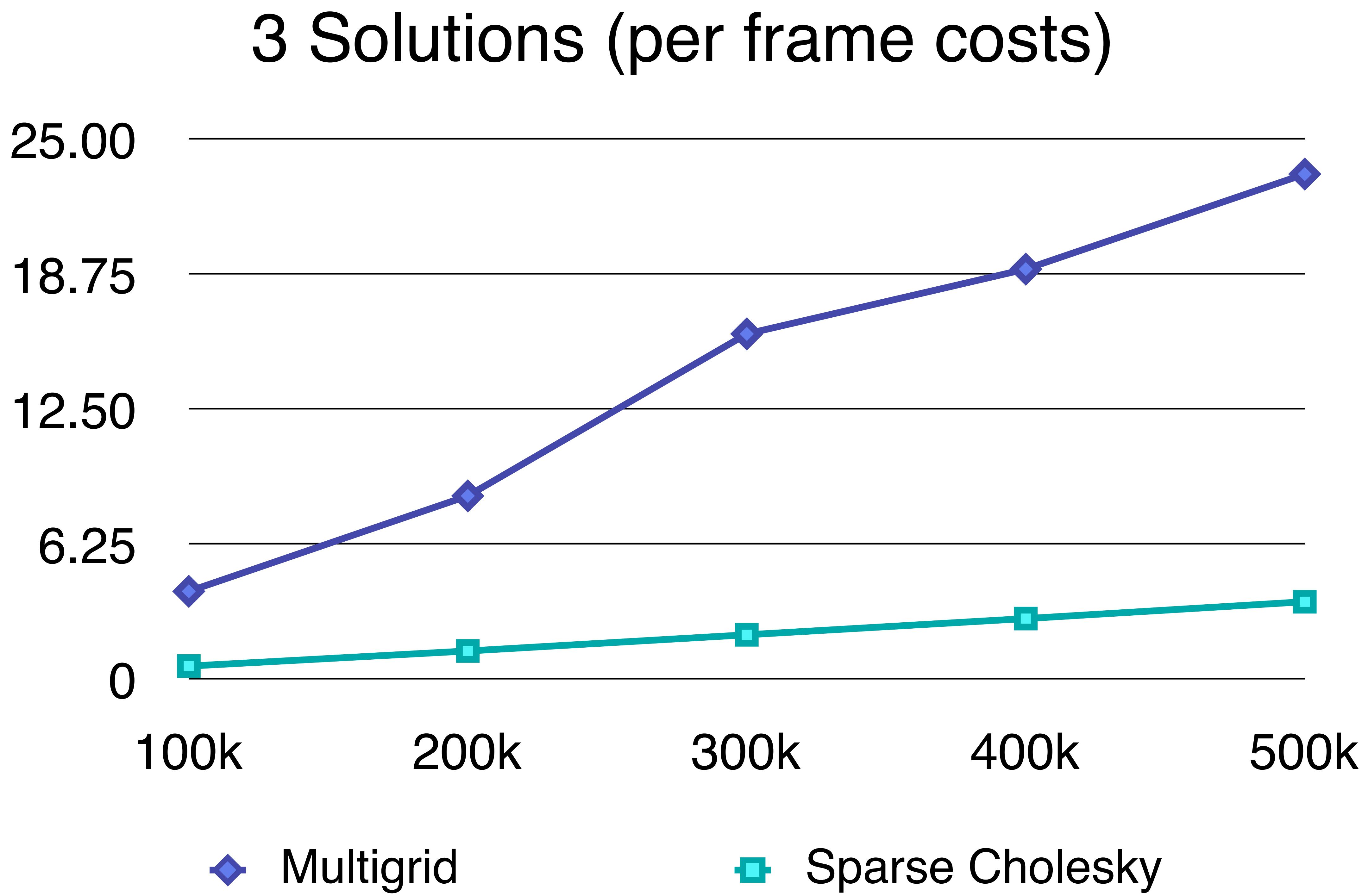
Small Bi-Laplace Systems



Large Bi-Laplace Systems

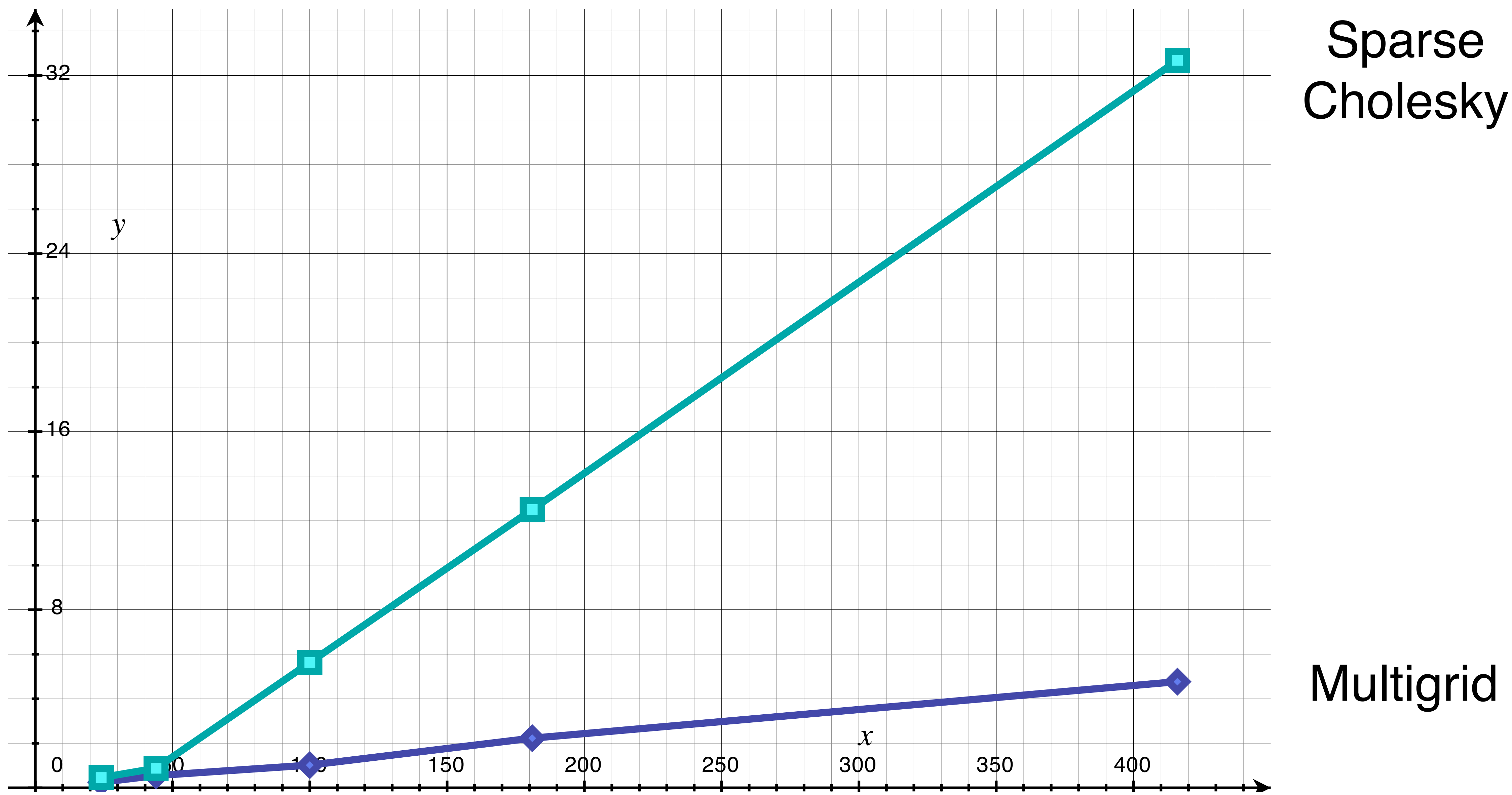


Large Bi-Laplace Systems



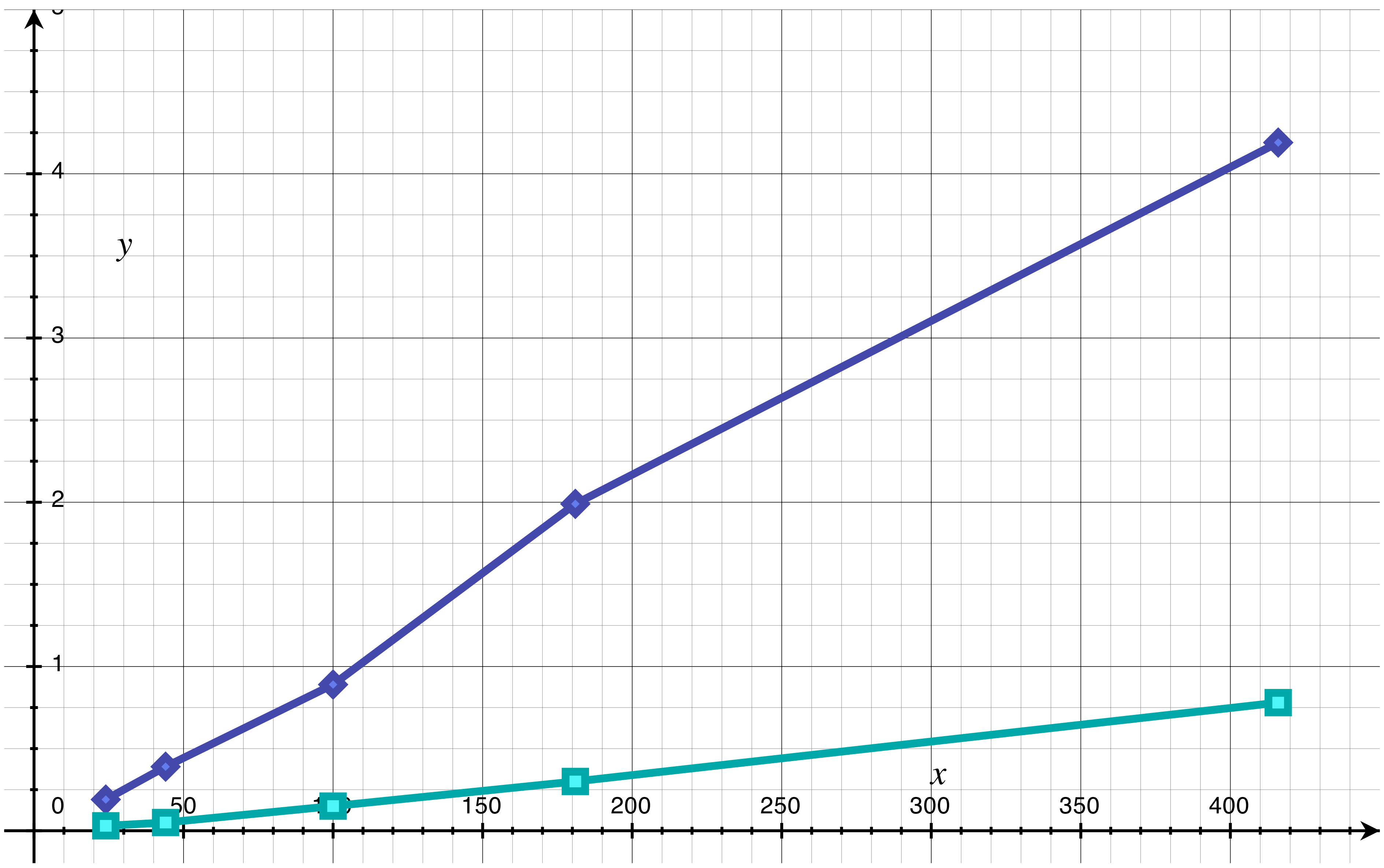
Shi et al., Fast MG Algo

Setup + Precomp. + 3 Solutions



Shi et al., Fast MG Algo

3 Solutions (per frame costs)



Multigrid

Sparse Cholesky

Conclusion

- Typical geometry processing problems are
  - large but sparse
  - symmetric positive definite
- Multigrid solvers
  - Require careful implementation
  - Use it if mesh / matrix changes frequently
- Direct sparse solvers
  - Easy to use (black-box)
  - Well suited for multiple rhs, or if only matrix values change